
175 PTAS
67
mi computer
CURSO PRACTICO DEL ORDENADOR PERSONAL,
EL MICRO Y EL MINIORDENADOR
Editorial Delta, S.A.

MI COMPUTER CURSO PRACTICO DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen VI-Fascículo 67

Director: José Mas Godayol
Director editorial: Gerardo Romero
Jefe de redacción: Pablo Parra
Coordinación editorial: Jaime Mardones, Francisco Martín
Asesor técnico: Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.°, 08008 Barcelona
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-7598-034-7 (tomo 6)
84-85822-82-X (obra completa)
Depósito Legal: B. 52/1984

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 248504

Impreso en España-Printed in Spain-Abril 1985

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 19 425 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 429 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S.A. (Paseo de Gracia, 88, 5.°, 08008 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

No se efectúan envíos contra reembolso.





Robots de élite

Fijemos nuestra atención en los robots más sofisticados que es posible encontrar en el mercado: los que se utilizan para enseñar los principios de la robótica y aquellos que son representativos del arte del diseño moderno de robots

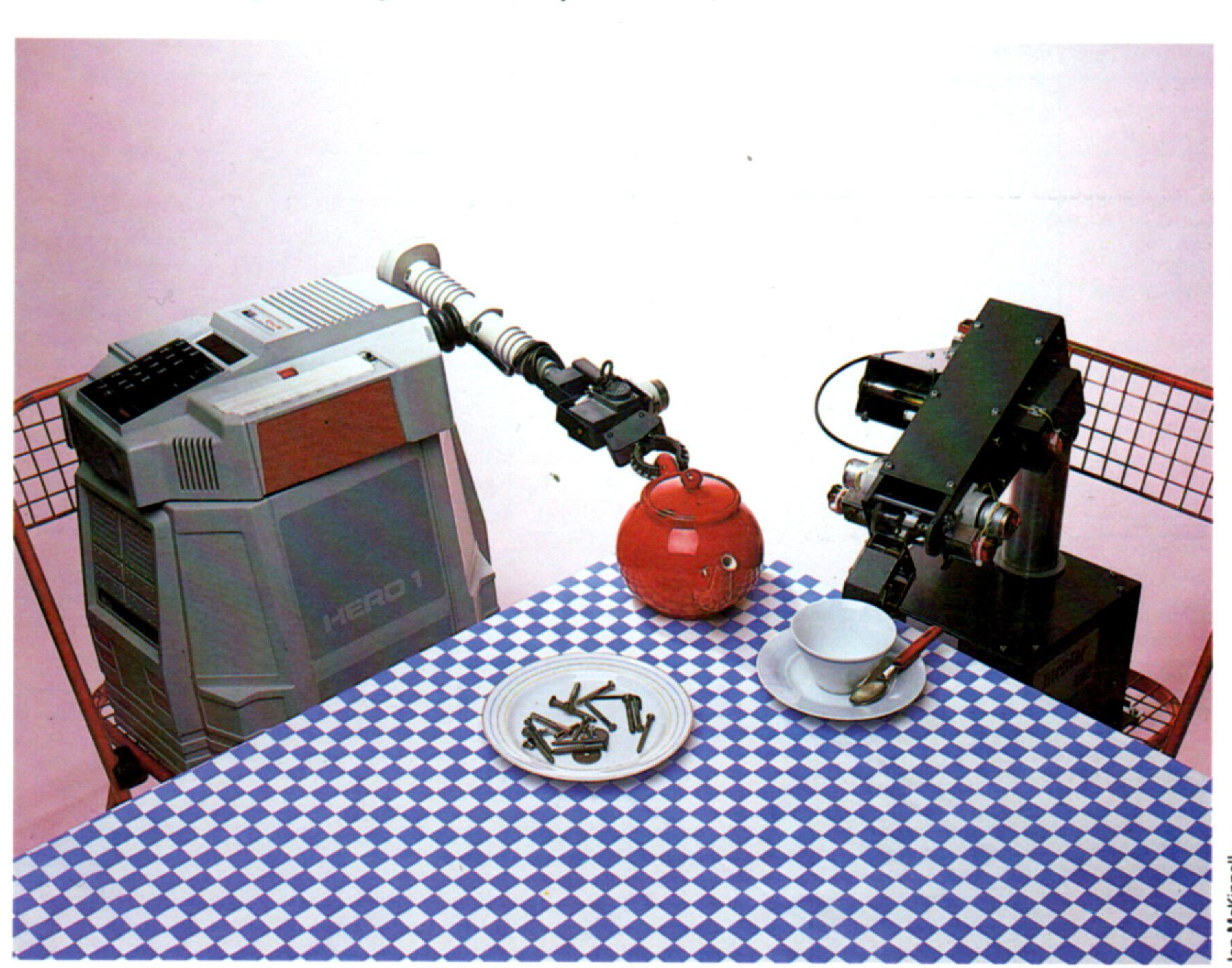
Muchos de los robots que vamos a ver en este capítulo son caros, pero no entran en la categoría de robots industriales y están diseñados para su uso tanto en el hogar como en la escuela. El primer grupo que analizaremos es el que engloba los robots cuya ingeniería responde a un estándar elevado e incorporan muchas de las características de los brazos-robot industriales. La principal diferencia entre éstos y los brazos industriales es que la mayoría de los que mencionaremos aquí están diseñados para uso didáctico y se emplean para enseñar los principios de la robótica.

La principal diferencia entre estos brazos y sus equivalentes industriales radica en que éstos son más pequeños y tienen menos capacidad de manipulación de objetos grandes. En muchos casos, por supuesto, el mercado educativo se superpone al industrial, ya que si lo que la aplicación industrial necesita es un brazo relativamente ligero y pequeño, muchos de estos brazos cumplen con tal condición. Por ejemplo, puede que se requiera un robot industrial para manipular grandes lingotes de acero que pesen cientos de kilos, o para realizar una labor muy diferente, como es ensamblar componentes en una placa de circuito impreso, tarea ésta para la que no se requiere un brazo grande y potente. Por lo tanto, se puede considerar que los brazos-robot de esta categoría poseen el potencial para llevar a cabo aplicaciones serias, además de ser didácticos.

La segunda categoría que veremos es la de aquellos robots cuyo diseño incorpora los últimos adelantos logrados en robótica. Muchos de ellos estarán equipados con los poderes sensoriales que ya hemos analizado en anteriores capítulos.

Robots didácticos

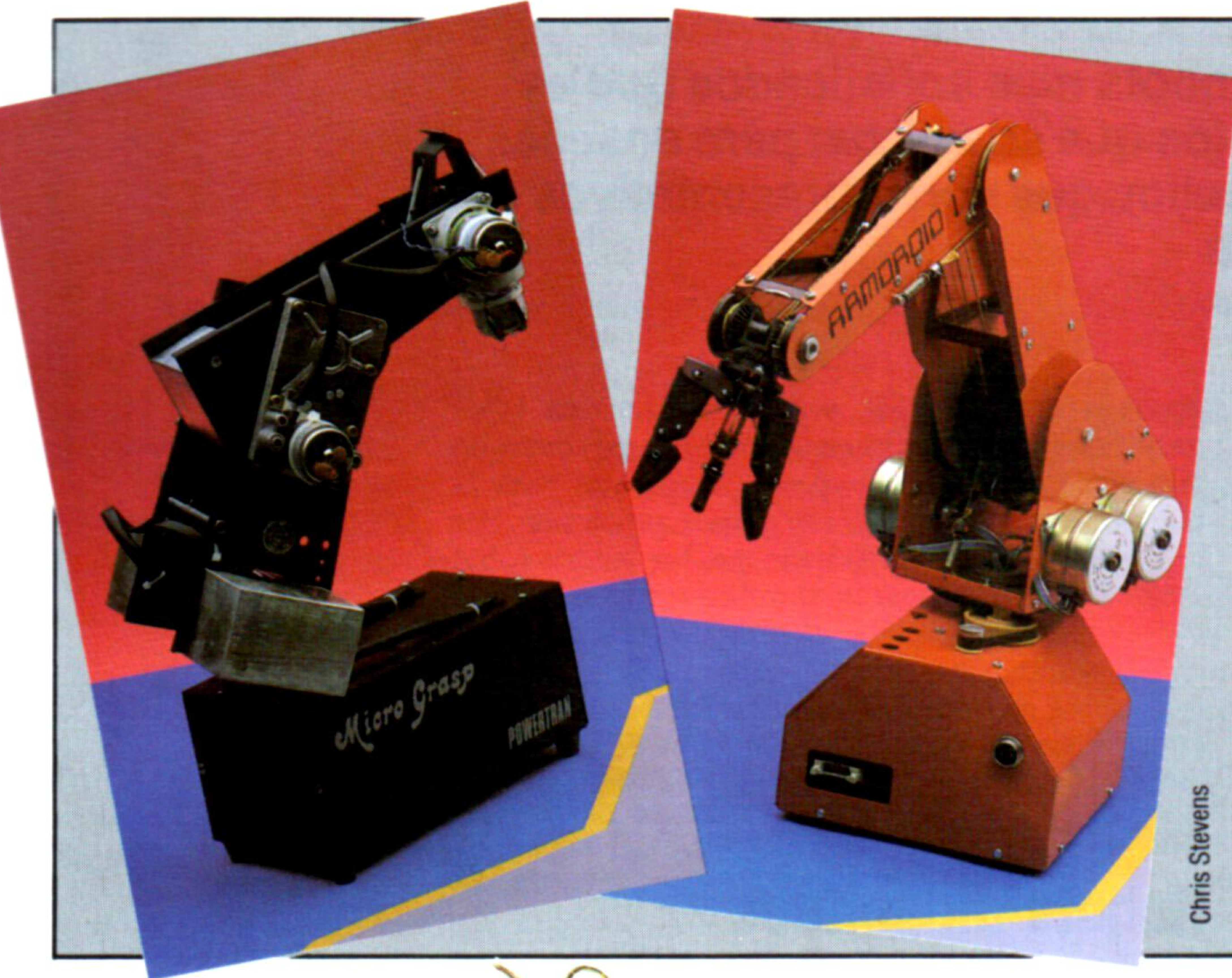
Un brazo-robot de precio moderado es el Mentor, de Cybernetic Applications. Se vende en forma de modelo para armar, posee seis grados de libertad (cintura, hombro, codo y tres ejes de rotación en la muñeca) y está activado por motores eléctricos. Se lo puede controlar desde un BBC Micro, un Vic-20 o un Spectrum.



En amable compañía
Puesto que el Hero es a la vez móvil y posee una pinza, puede servirle a su dueño una taza de té por las mañanas (¡siempre y cuando, por supuesto, no haya que subir alguna escalera desde la cocina hasta el dormitorio!). En la fotografía vemos al Hero y al Mentor disponiéndose a saborear una taza de té

Ian McKinnell

La misma empresa produce, a un precio muy superior, el Neptune 1 y el Neptune 2. Estos brazos pueden levantar hasta 2,5 kg y funcionan con energía hidráulica, aunque utilizan agua y no el fluido hidráulico normal que emplean la mayor parte de los robots de esta clase. Estos brazos también se



Brazos para aprender

Las dos áreas más significativas de la investigación en microelectrónica son el diseño de chips y la robótica. Son pocos los aficionados que pueden experimentar en los talleres de sus hogares con nuevos microchips, pero los robots se están volviendo cada vez más accesibles. La finalidad principal de los brazos-robot reseñados en este capítulo no es otra que contribuir a una mayor comprensión del usuario acerca de cómo funcionan los robots. La mejor forma de acrecentar nuestros conocimientos sobre éstos es examinar sistemas de operación e intentar perfeccionarlos. Los brazos-robot que vemos en la fotografía, el Micro Grasp de Powertran Cybernetics y el Armdroid de Colne Robotics, se programa mediante el uso de microordenadores y poseen cinco grados de libertad. Se los puede adquirir totalmente montados o en forma de modelos para armar

Chris Stevens

NOMBRE	NEPTUNE 1	NEPTUNE 2	MENTOR	GENESIS P101	HERO
TIPO	Brazo	Brazo	Brazo	Brazo	Robot móvil
APLIC. PRINCIPAL	Didáctica	Didáctica	Didáctica	Didáctica	Didáctica/experimental
SENSORES	El potenciómetro registra la posición, la pinza puede detectar su grado de cierre	El potenciómetro registra la posición, la pinza puede detectar su grado de cierre	El potenciómetro registra la posición, la pinza puede detectar su grado de cierre	Realimentación posicional	Ultrasónico, permite la detección de movimiento. Los sensores detectan 256 niveles de luz y sonido. Sensores táctiles en la pinza
GRADOS DE LIBERTAD	6: cintura, hombro, codo y elevación de muñeca, tonel de muñeca y pinza	7: cintura, hombro, codo, elevación de muñeca, tonel de muñeca, guiñada de muñeca y pinza	6: cintura, hombro, codo y elevación de muñeca, tonel de muñeca y pinza	6: cintura, hombro, codo y elevación de muñeca, tonel de muñeca y pinza	4: hombro, codo, muñeca y pinza
PRECIO*	£ 1 250	£ 1 725	£ 345	£ 1 750	£ 1 995
ENERGIA	Hidráulica: bomba de agua activada por corriente eléctrica	Hidráulica: bomba de agua activada por corriente eléctrica	Eléctrica	Hidráulica: bomba de agua activada por corriente eléctrica	Baterías recargables
SE CONECTA AL	BBC Micro, Spectrum, Vic-20	BBC Micro, Spectrum, Vic-20	BBC Micro, Spectrum, Vic-20	Programado a través de caja controladora; interface estándar RS232; BBC Micro, Spectrum, Vic-20	Utilizando un ensamblador cruzado, el HERO se puede conectar a cualquier micro que posea una puerta en serie
FABRICADO POR	Cybernetic Applications Ltd., West Portway Industrial Estate, Andover, Hampshire SP10 3NN	Cybernetic Applications Ltd.	Cybernetic Applications Ltd.	Powertran Cybernetics, West Portway Industrial Estate, Andover, Hampshire SP10 3NN	Zenith Data Systems, Bristol Road, Gloucester, GL2 6EE

*En el mercado británico

pueden controlar con el BBC Micro, el Vic-20 y el Spectrum. El Neptune 2 tiene dos velocidades de operación diferentes; esto es de gran utilidad, porque se lo puede hacer mover rápidamente cuando se requieren grandes desplazamientos del brazo, y hacer luego que reduzca su velocidad para los movimientos que exijan mayor precisión.

Powertran Cybernetics produce el Genesis P101, que posee seis grados de libertad y se vende en forma de modelo para armar. Este modelo funciona con energía hidráulica y viene con una caja controladora para programar el robot, junto con una interface RS232 estándar gracias a la cual se lo puede conectar a la mayoría de los ordenadores. Este brazo también se puede adquirir premontado y probado a un precio superior.

Un interesante brazo de precio medio es el Cyber 310 de Cyber Robotics. Se vende premontado. Existen versiones para el BBC Micro, el Jupiter Ace, el Apple II, los Commodore Pet 3000/4000 y 8000 y el Hector HRX. Funciona mediante motores paso a paso y su capacidad para levantar pesos es de sólo 250 g, por lo cual es bastante ligero, pero la gama de opciones que ofrece es notable. Además de disponer de cinco grados de libertad, permite que el propio usuario controle la aceleración y deceleración del brazo, lo que significa que puede imitar la forma en la cual se mueve un brazo humano, alterando constantemente su velocidad según vaya cambiando la naturaleza de la tarea e imitando los efectos de la inercia. Todas las junturas se pueden mover simultáneamente y es posible especificar la posición del brazo ya sea como una posición relativa (p. ej., indicándole al brazo que se desplace *x* unidades hacia adelante respecto a su posición actual) o bien como una posición absoluta (especificando un movimiento hasta algún punto en relación a una posición "base"). Puede ser programado en BASIC y también en una versión de FORTH, conocida como Robo FORTH y desarrollada por la firma fabricante.

Ascendiendo en la gama de precios llegamos al HRA 933 y HRA934 de Feedback Instruments, que se venden ya montados. Ambos son brazos que funcionan con energía hidráulica y pueden levantar 1,35 kg con una precisión de posicionamiento de 3 mm. Además de contar con sensores de posición para las junturas del brazo, los brazos poseen sensores táctiles en sus efectores finales. Estos sensores indican que han cogido algo y permiten que el brazo controle la fuerza aplicada en el momento de recoger objetos. El control se realiza a través de una interface RS232 y las instrucciones específicas para el control se imparten utilizando el Apple II, el Tandy TRS-80, el Commodore PET, el AIM 65 y el MAT385.

CYBER 310	HRA933	HRA934
Brazo	Brazo	Brazo
Didáctica	Didáctica	Didáctica
Ninguno	Capacidad para determinar su posición, la pinza puede detectar su grado de cierre	Capacidad para determinar su posición, la pinza puede detectar su grado de cierre
6: rotación de base, rotación de hombro (que se puede inclinar en 180°), codo, elevación de muñeca y torsión de muñeca y pinza	5: rotación de base, codo, tonel de muñeca, guiñada de muñeca y pinza	5: rotación de base, codo, tonel de muñeca, guiñada de muñeca y pinza
£ 650	£ 2 195	£ 2 726
Eléctrica	Eléctrica	Eléctrica
BBC Micro, Jupiter Ace, Apple II, serie Commodore PET, Hector HRX	Apple II, Tandy TRS80, serie Commodore PET, AIM 65, BBC Micro, MAT 385	Apple II, Tandy TRS80, serie Commodore PET, AIM 65, BBC Micro, MAT 385
Cyber Robotics, 61 Ditton Walk, Cambridge, CB5 8QD	Feedback Systems Ltd., Park Road, Crowborough, East Sussex, TN6 2QR	Feedback Systems Ltd.

La robótica en estos momentos

El robot Hero-1 de Zenith Data Systems se vende en forma de modelo para armar y ofrece algunas facilidades bastante notables. Es móvil y posee un brazo que hace uso de un sistema de coordenadas esféricas en virtud del cual el brazo puede extenderse y contraerse telescópicamente. El Hero está equipado con una gran matriz de sensores para detectar movimiento, sonido y luz, incluyendo un sensor ultrasónico de distancia que contribuye a evitar colisiones, y un sintetizador de voz que le proporciona un vocabulario ilimitado. Asimismo, posee un brazo con cinco grados de libertad. El ensamblaje es muy laborioso; por este motivo es posible que el interesado desee adquirirlo ya armado. En este caso el precio se eleva sustancialmente.

Todos estos robots son, en cierto sentido, poco más que un entretenimiento de lujo, desde el punto de vista de lo que realmente pueden hacer para el usuario; y, en realidad, hasta el momento su principal uso ha sido por parte de firmas comerciales que desean emplear un robot con fines publicitarios: repartir en mano folletos en los puestos de las ferias de muestras o hacer demostraciones de productos. No obstante, representan la mejor tecnología de la robótica existente en estos momentos. Todos ellos utilizan sensores de una forma inteligente, se desplazan inteligentemente y poseen brazos inteligentes. Ninguno de ellos dispone de algún sistema de visión, pero sí pueden hablar y pueden oír señales acústicas y responder a las mismas.

Huelga decir que su precio disuadirá a muchas personas de la tentación de comprarlos, pero no por ello dejan de estar allí, como algo a lo que aspirar, algo que quizá usted mismo sea capaz de hacer realidad construyéndose su propio robot. Asimismo, constituyen un indicio del ritmo al cual está avanzando la robótica.

Despegue vertical

Iniciamos una serie en que analizaremos el denominado "software vertical", desarrollado para cumplir una función específica relacionada con medicina, derecho, periodismo, fotografía y otras actividades especializadas

Existen muchísimas aplicaciones para los ordenadores personales, muchas de las cuales no son evidentes. Estas aplicaciones se han desarrollado a consecuencia de la adaptación creativa de software ya existente o de la generación de software para uso especial. Se dice que éstas son aplicaciones del "mercado vertical" porque se aplican a un grupo específico de personas, como pueden ser médicos, químicos o psicólogos. En esta serie analizaremos numerosos paquetes para el mercado vertical que revelan nuevas e interesantes facetas del uso de los microordenadores. Los siguientes ejemplos ayudarán a ilustrar la clase de problemas para cuya solución se diseña software vertical.

Un grupo de padres de adolescentes adictos a la heroína, residentes en el West End londinense, está utilizando el programa *BrainStorm*, de Caxton Software, para planificar una campaña de divulgación de sus actividades entre los médicos, asistentes sociales y órganos para la aplicación de la ley de la localidad. El dueño de un restaurante de Kent está utilizando una hoja electrónica *Practicalc* para analizar los pedidos de los clientes, con el fin de planificar sus futuros menús. Una granja de Sussex está utilizando el mismo programa (trabajando con cuatro Commodore 64) para manejar todo, desde sus proyectos de gran importancia hasta el control de sus costos de energía. Un cirujano de un hospital de Londres está aplicando el programa *Superbase*, de Precision Software, en la investigación que está llevando a cabo sobre las causas del cáncer y su curación.

En Gran Bretaña hay médicos que están empleando ordenadores para cumplir con la nueva normativa de que todas las etiquetas de las recetas de los pacientes deben ir impresas y no escritas a mano. Un diseñador de cocinas, de Lancashire (Gran Bretaña), utiliza un programa (desarrollado para un BBC Micro por una pareja de amigos con los que juega al billar) para ir colocando hornos, neveras y otros artefactos de cocina en diferentes sitios sobre un plano del espacio disponible basado en pantalla. De hecho, le ha resultado una ayuda tan eficaz en su negocio de muebles de cocina (y dormitorio), que ha formado, junto con los diseñadores originales del programa, una sociedad para ofrecerles el sistema a otros comerciantes.

Éstos son apenas unos pocos ejemplos de las nuevas respuestas a la ya tradicional pregunta que formula todo probable comprador de un ordenador nuevo: "¿Para qué sirve?" Se ha calculado que el usuario medio de un ordenador no explota más del 10 % del potencial de la máquina, y ésta puede ser una estimación incluso optimista. Invertir dinero en algo que luego se limita en un uso final, ya sea entretenerse con juegos o administrar las cuentas, no es tan rentable, teniendo en cuenta el costo, como explotar su versatilidad al máximo.

En este sentido existen dos alternativas. Una opción es encontrar nuevas aplicaciones para el software estándar. Un fotógrafo profesional de Sheffield utiliza el módulo de control de stocks del paquete *Anagram integrated accounts* para administrar su biblioteca de fotografías, con lo que consigue sacar rendimiento de los originales de su archivo de copias y transparencias antiguas. Del mismo modo, algunas agencias de colocaciones utilizan la base de datos *Tomorrow's office* para cotejar las necesidades de sus clientes con el potencial humano disponible, conservando los datos de los *curriculum vitae* en disco rígido y enviándolos por correo automáticamente. Esto es mucho más barato que utilizar paquetes diseñados específicamente para esa tarea: el paquete *Body matching and marketing*, producido por la firma AP Computer Consultants, tiene un precio que sobrepasa las mil libras esterlinas (unas 200 000 ptas.).

Diseño a medida

La alternativa es buscar un paquete diseñado para el uso específico que uno necesite. Imagínese que es un estudiante de teología y tiene la mesa de estudio abarrotada de tomos enormes, concordancias, notas, diccionarios bíblicos y cosas por el estilo. Bueno, puede que lo que necesite sea el *"The Word" processor* (procesador de la Palabra), de Bible Research Systems para el IBM y máquinas compatibles. Este paquete incluye la traducción completa del rey Jacobo I, con facilidades completas de búsqueda para la creación de referencias cruzadas en disco.

Y si no se fía de ninguna de las traducciones del material bíblico, puede compararlas con el original mediante un programa llamado *The Greek transliterator*, que le dará el equivalente en griego de cualquier palabra o frase en inglés, y visualizará todos los casos en los que aparezca en el texto en inglés. Esto permite comparar las diversas formas en que se ha traducido una palabra. Estos estudios electrónicos de la Biblia, sin embargo, no son baratos. Estos dos programas le supondrán un gasto que ascenderá a las 253 libras esterlinas cada uno (unas 50 000 ptas.).

Vamos a ver otro paquete que a primera vista parecería igualmente improbable de hallar. Si usted es un ingeniero que está planificando desagües y alcantarillas, entonces el *MIDDUSS* (el *McMaster interactive design of stormwater systems*) le ayudará a medir tuberías, canales y estanques de almacenamiento, permitirá que genere hidrográficos y le proporcionará vuelcos en pantalla de todo su tra-

bajo en gráficos en alta resolución. Para ejecutar todo esto necesitará un Sirius de 256 K.

Los sistemas de alcantarillado parecen haber proporcionado a los programadores muchísimo estímulo. El robusto ordenador de mano Husky es el corazón del programa *CAMIL (Computer-aided manhole inspection and location)*, desarrollado en Southampton y que ahora están aplicando en toda Gran Bretaña las autoridades de aguas. El programa permite que los inspectores entren datos en el campo y los mismos sean transmitidos a través de líneas telefónicas a los ordenadores centrales, que pueden entonces imprimir trazados de las cloacas. El programa puede incluso responder a una petición para ver "todas las cloacas de ladrillo construidas antes de 1900"; y de éstas hay una cantidad abrumadora.

A los viajantes de comercio les resultará sumamente útil la serie de programas "Travelling" que se ejecutan en otro ordenador de mano: el excelente NEC PC-8201. Además de paquetes básicos como el *Travelling writer* (escritor viajante: un procesador de textos para redactar informes, con capacidades para correo electrónico y administración de datos), la serie incluye el *Travelling project manager*, el *Travelling appointment manager*, el *Travelling sales manager* y (el más esencial) el *Travelling expense manager* (administrador de proyectos, de citas, de ventas y de gastos, respectivamente).

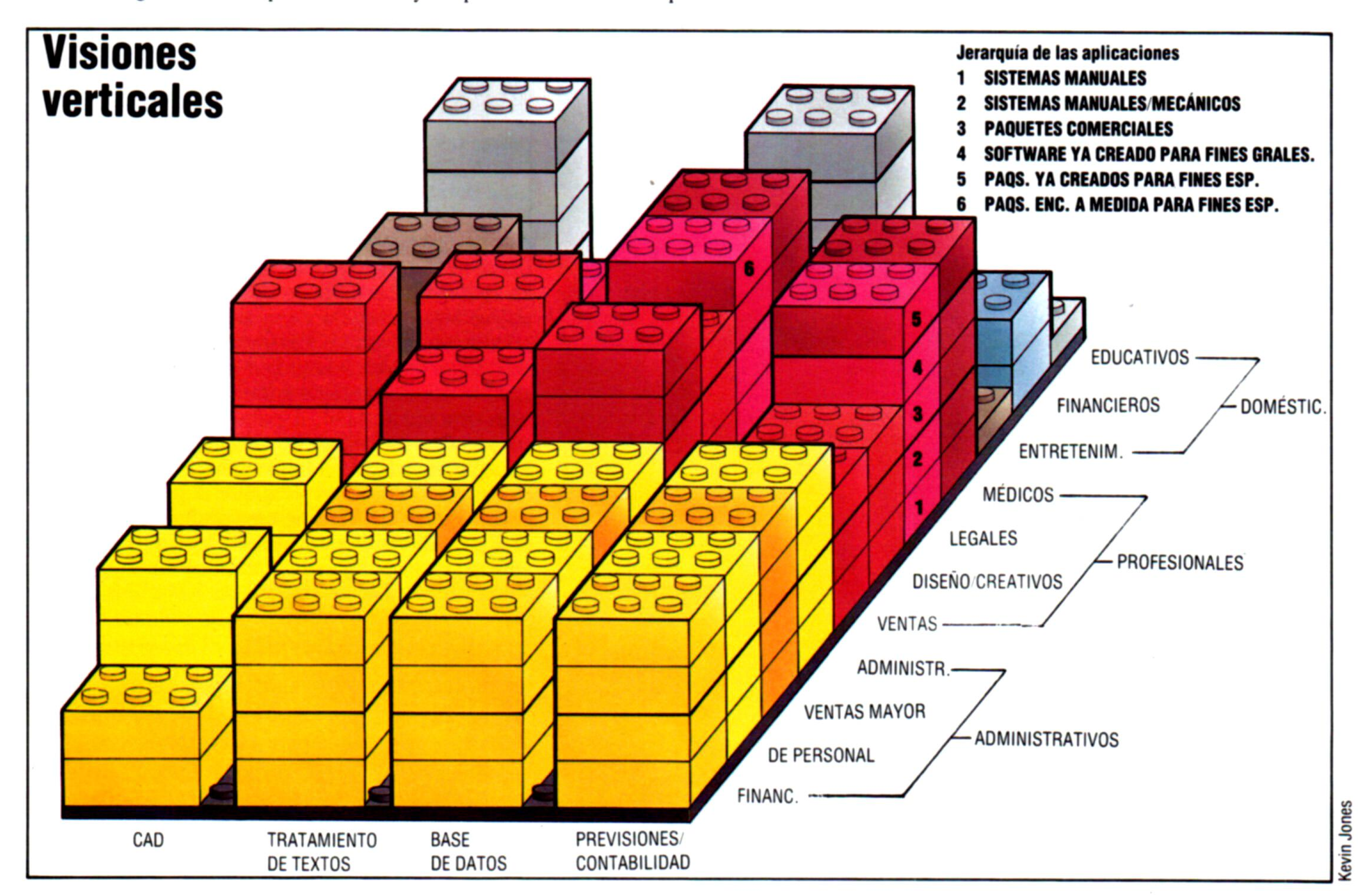
Los vendedores también pueden mejorar sus técnicas de venta con el módulo Sales Edge del juego de programas *Human edge*. Éstos son distribuidos por Thorn EMI para el IBM y el Apricot. El módulo Sales Edge evalúa los puntos fuertes y los puntos débiles del usuario de cara a la venta mediante una serie de preguntas y respuestas del tipo acuerdo/desacuerdo; tras una serie similar de preguntas al cliente, sugiere una estrategia de venta con maniobras de "apertura" y "cierre".

Si piensa invertir en divisas, quizá sienta la tentación de adquirir el Forextend de Forexia, que le permite estudiar y analizar lo que está sucediendo con el dólar, la libra esterlina, el franco suizo, el yen japonés y el marco alemán. El programa produce 37 gráficas de comparaciones, indicadores relativos de robustez, tasas de interés e índices de influencia comercial para cada día del período entre el 1 de octubre de 1983 y el día actual.

Los programas de citas y horarios están, asimismo, adquiriendo gran popularidad. La mayoría de los ordenadores tienen relojes incorporados, pero pocos de ellos pueden interrumpir cualquier actividad que el usuario esté desarrollando para recordarle dónde debería estar en ese momento. No obstante, el planificador de horarios basado en ROM de Hewlett-Packard para su HP-75C hace exactamente eso y, además, con toda una variedad de sonidos de aviso diferentes.

En el próximo capítulo de esta serie analizaremos en profundidad el *BrainStorm*, un programa del cual se afirma que organiza sus pensamientos de la misma manera en que un procesador de textos organiza sus oraciones. Y, por consiguiente, analizaremos paquetes de software para personas dedicadas al comercio, la medicina, la educación, la investigación, el periodismo, el derecho, el video, los espectáculos y la publicidad, con casos concretos de usuarios específicos.

Perfiles de mercado
El software se escribe para satisfacer las necesidades de grupos de usuarios primarios y para venderlo en los sectores fundamentales del mercado. El grueso del mismo, por lo tanto, comprende en gran parte paquetes financieros, de tratamiento de textos y de base de datos; una cuarta área, la del diseño asistido por ordenador (CAD: *Computer-aided design*), está creciendo en importancia. En el diagrama, estas áreas de aplicaciones se yuxtaponen a una selección de usuarios. El eje vertical representa el nivel de complejidad de la aplicación, estando los métodos manuales en el nivel inferior, el software para ordenador de uso general en el nivel medio, y el software diseñado a medida o con fines específicos en el nivel superior. Los perfiles de uso resultantes muestran que el software para fines generales suele adaptarse bien a las exigencias empresariales, pero que los usuarios necesitan adaptar a su medida los paquetes o bien crearse los suyos; los perfiles alto-bajo irregulares indican una descompensación entre las necesidades y los recursos, los perfiles de altura media regulares muestran la capacidad del mercado para equilibrar necesidades y clientes



Tomar y llevar

Prosiguiendo con nuestro juego, desarrollaremos las rutinas necesarias para recoger y transportar objetos entre escenarios

En el último capítulo de este proyecto vimos el análisis de las instrucciones y diseñamos un grupo de instrucciones "normales". En este grupo incluimos las instrucciones RECOGER y DEJAR, junto con sus variantes COGER y PONER. Una vez reconocida la instrucción apropiada, podemos construir las rutinas que obedezcan la instrucción. En primer lugar vamos a considerar RECOGER.

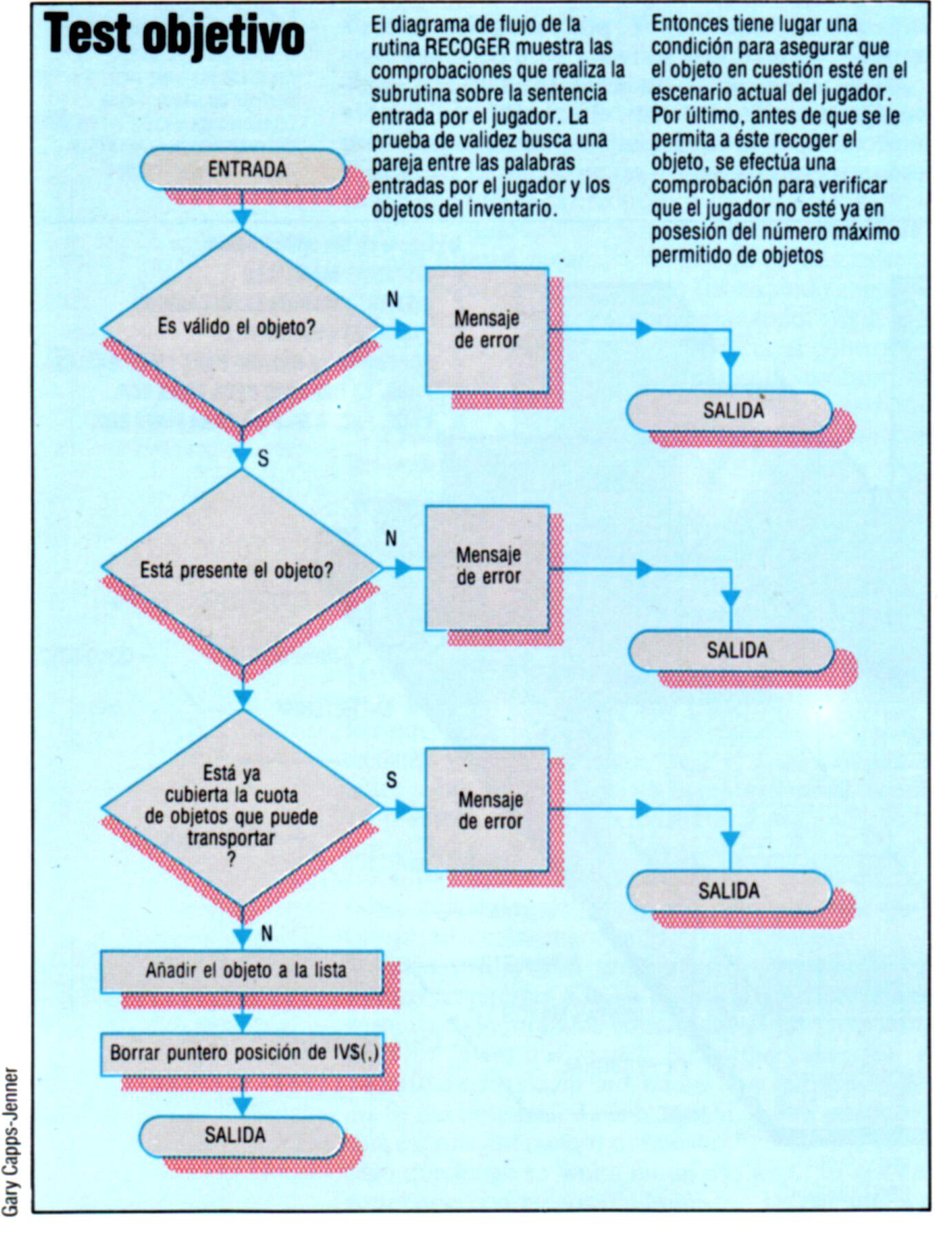
Para entender los métodos que emplea la rutina RECOGER, resumamos la forma en que el programa lleva el registro de los objetos de nuestro mundo de aventuras. En el primer capítulo del proyecto diseñamos sentencias DATA para cada escenario que contenían descripciones del mismo, los nombres de los objetos presentes e información sobre las salidas posibles. Después de leer los datos, la matriz IV$(,) (utilizada para almacenar los datos de los objetos de *El bosque encantado*) posee el siguiente contenido:

N	IV$(N,1)	IV$(N,2)
1	ESCOPETA	10
2	FAROL	9
3	LLAVE	5

La primera columna de la matriz contiene el nombre del objeto, mientras que la segunda contiene el número de su escenario inicial en el mapa del mundo de aventuras. Durante la descripción de cualquier escenario, se explora la segunda columna de esta matriz para ver si cualquiera de los objetos se halla en el escenario actual del jugador, P. Cuando éste desea recoger un objeto de un escenario, utilizando una instrucción ajustada al formato RECOGER EL OBJETO, se deben considerar varios factores:

- ¿Es válido el objeto de la instrucción?; en otras palabras, ¿aparece en la matriz del inventario, IV$(,)?
- ¿Está presente el objeto en el escenario actual del jugador?
- ¿Posee ya éste la cuota completa de objetos que permiten las reglas del juego?

Si se puede responder satisfactoriamente a estas preguntas, entonces el jugador puede recoger el objeto. Esto implica añadir la descripción del objeto a la matriz de objetos personales del jugador, IC$(), y borrar el indicador de posición de la entrada correspondiente en IV$(,). Observe que el nombre del objeto no se tiene que borrar. Si utilizamos un indicador de posición de −1 para cada objeto que se haya recogido y transportado, tales objetos no aparecerán en las descripciones de los escenarios. Sería muy curioso coger la ESCOPETA en el escenario 10, desplazarse hasta el escenario 9 y después otra vez hasta el 10, para encontrarse al regreso con que la ESCOPETA todavía está allí. Por lo tanto, la matriz IV$(,) lleva un registro de las posiciones de todos los objetos que el jugador *no* está llevando consigo. El diagrama de flujo para la rutina RECOGER muestra la sencilla lógica que se debe aplicar.



Test objetivo

El diagrama de flujo de la rutina RECOGER muestra las comprobaciones que realiza la subrutina sobre la sentencia entrada por el jugador. La prueba de validez busca una pareja entre las palabras entradas por el jugador y los objetos del inventario. Entonces tiene lugar una condición para asegurar que el objeto en cuestión esté en el escenario actual del jugador. Por último, antes de que se le permita a éste recoger el objeto, se efectúa una comprobación para verificar que el jugador no esté ya en posesión del número máximo permitido de objetos

Gary Capps-Jenner

```
3700 REM **** S/R RECOGER ****
3710 GOSUB 5300:REM ES VALIDO EL OBJETO
3720 IF F=0 THEN SN$="NO HAY NINGUN "+W$:GOSUB5500:
     RETURN
3730 OV=F:GOSUB5450:REM COMPROBAR INVENTARIO
3740 IF HF=1 THEN SN$="TU YA LLEVAS "+IV$(F,1):GOSUB 5500:
     RETURN
3750 :
3755 REM ** ESTA AQUI EL OBJETO ? **
3760 IF VAL(IV$(F,2))<>P THEN SN$=IV$(F,1)+" NO ESTA
     AQUI":GOSUB5500:RETURN
3770 :
3780 REM ** AÑADIR OBJETO A LA LISTA **
3790 A=0
3800 FOR J=1 TO 2
```



```
3810 IF IC$(J)="" THEN IC$(J)=IV$(F,1):AF=1:J=2
3820 NEXT J
3830 :
3840 REM ** CUOTA COMPLETA **
3850 IF AF=0 THEN PRINT"YA TIENES DOS OBJETOS": RETURN
3860 :
3870 SN$="RECOGES EL "+IV$(F,1):GOSUB5500
3880 IV$(F,2)="-1":REM BORRAR ENTRADA INVENTARIO
3890 RETURN
```

Ahora vamos a analizar cada una de estas tres condiciones por separado.

La prueba de validez

De las tres comprobaciones, la más complicada e importante es la *prueba de validez*. En su forma más simple, ésta podría ser una rutina que se limitara a tomar la segunda parte de la instrucción descompuesta y compararla con cada uno de los elementos de la matriz del inventario, IV$(,). Sin embargo, si fuera éste el caso, la instrucción RECOGER se vería limitada a la estructura rigurosa de RECOGER OBJETO. Incluso variaciones tales como RECOGER LA ESCOPETA serían inaceptables, puesto que la rutina intentaría emparejar LA ESCOPETA con el inventario, en vez de emparejar sólo ESCOPETA. Para permitir cierta flexibilidad en la estructura de la instrucción RECOGER hemos de desarrollar un método más sofisticado para comparar la segunda parte de la instrucción dada con el inventario de objetos.

El procedimiento más evidente para aumentar la flexibilidad consiste en dividir la segunda parte de la instrucción dada en las palabras que la componen y luego comparar cada una de ellas con el inventario de objetos. Si bien con ello solucionaríamos el problema esbozado anteriormente, este método también tiene sus inconvenientes. Si, por ejemplo, quisiéramos utilizar una descripción de dos palabras para un objeto, como CUCHILLO GRANDE, entonces utilizando este método la instrucción RECOGER EL CUCHILLO GRANDE no tendría pareja. La rutina compararía las palabras EL, CUCHILLO y GRANDE por separado con la lista del inventario. Este problema se puede solventar haciendo que la rutina sea aún más refinada. En vez de buscar una pareja exacta, se podría diseñar una rutina que explorara cada descripción de objetos del inventario para la palabra de instrucción que se esté examinando, desplazándose letra por letra a través del nombre del objeto hasta encontrar una pareja o hasta llegar al final del nombre del objeto. La pantalla muestra cómo se realiza esto.

Una ventaja que ofrece el buscar una pareja entre instrucción e inventario de esta manera, es que se pueden dar en la instrucción versiones abreviadas de la palabra objeto. En el ejemplo de arriba, la instrucción RECOGER EL CUCH también formaría la pareja correcta, suponiendo que antes de CUCHILLO GRANDE no hubiera en el inventario ningún otro nombre de objeto que tuviera la combinación de letras CUCH. De ser así, entonces se formaría una pareja incorrecta con la entrada anterior del inventario. Los problemas de este tipo forman parte del precio que se debe pagar por la mayor flexibilidad de la rutina. La mayoría de los problemas de emparejamiento incorrecto se pueden eliminar mediante una cuidadosa selección de los nombres de los objetos. Si dos nombres de objeto deben contener el mismo grupo de caracteres, o si un nombre es una subserie de otro (como SOL y

La gran coincidencia

```
  10 REM **** DEMOSTRACION EMPAREJAR UN OBJETO ****
  40 MODE 5:COLOUR 2:DIM V$(3)
  60 FOR I=1 TO 3:READ V$(I):NEXT I
  90 A$="THE":B$="KNIFE":C$="KNI"
  95 S$="              "
 110 M$=A$:GOSUB 1000:REM EMPAREJAR "THE"
 130 M$=B$:GOSUB 1000:REM EMPAREJAR "KNIFE"
 150 M$=C$:GOSUB 1000:REM EMPAREJAR "KNI"
 160 END
1000 REM **** S/R EMPAREJAR ****
1010 CLS:F=0:LW=LEN(M$)
1030 FOR J=1 TO 3:LI=LEN(V$(J))
1042 X=1:Y=6:GOSUB 2000:PRINT S$
1045 X=1:Y=6:GOSUB 2000:PRINT V$(J)
1047 FOR I=1 TO LI-LW+1
1050 X=1:Y=5:GOSUB 2000:PRINT S$
1060 X=I:Y=5:GOSUB 2000:PRINT M$
1070 IF MID$(V$(J),I,LW)=M$ THEN F=I:I=LI:J=3
1075 FOR D=1 TO 300:NEXT D,I:REM DEMORA
1086 IF F<>0 THEN GOSUB 2500
1087 NEXT J:RETURN
2000 REM **** POSICIONAR CURSOR EN X,Y ****
2010 PRINT TAB(X,Y);:RETURN
2500 REM **** HALLADA PAREJA ****
2502 COLOUR 1:X=F:Y=6:GOSUB 2000:PRINT M$
2505 FOR K=1 TO 5:X=1:Y=10:GOSUB 2000:PRINT S$
2508 FOR D=1 TO 500:NEXT D:REM DEMORA
2510 X=1:Y=10:GOSUB 2000:PRINT "MATCH FOUND":REM
     "HALLADA PAREJA"
2512 FOR D=1 TO 500:NEXT D,K:REM DEMORA
2520 A$=GET$:COLOUR 2:RETURN
3000 REM **** DATOS INVENTARIO ****
3005 DATA "SMALL FORK","RED DOOR","LARGE KNIFE"

  10 REM ** PAREJA SPECTRUM **
  40 INK 6:DIM V$(3,20)
1070 IF V$(J,I TO I+LW-1)=M$ THEN F=I:I=LI:J=3
2010 PRINT AT (Y,X);:RETURN
2502 INK 2:X=F:Y=6:GOSUB2000:PRINT S$
2520 A$=INKEY$:IF A$="" THEN 2520
2525 INK 6:RETURN

  10 REM ** PAREJA CBM 64 **
  40 PRINT CHR$(158):DIM V$(3)
  50 DN$=CHR$(17):FOR K=1 TO
     5:DN$=DN$+DN$:NEXT:DN$=CHR$(19)+DN$
2010 PRINT LEFT$(DN$,Y)TAB(X);:RETURN
2502 PRINTCHR$(28):X=F:Y=6:GOSUB2000:PRINT S$
2520 GETA$:IF A$="" THEN 2520
2525 PRINT CHR$(158):RETURN
```

La subrutina de prueba de validez diseñada para utilizar junto con la rutina RECOGER explora la sentencia entrada de palabra en palabra, tratando de hallar una pareja con una entrada del inventario. Este breve programa ilustra la forma en que la rutina de validez va buscando una pareja a lo largo del inventario. Para esta demostración, en el inventario hay tres objetos, y el programa intenta emparejar las palabras "THE", "KNIFE" y "KNI". Cada vez que se encuentra una pareja, el programa espera que se pulse una tecla antes de seguir adelante

SOLDADO), entonces el más corto de los dos nombres se debe colocar en un lugar anterior en la matriz del inventario. Además, no se deben utilizar descripciones de objetos diferentes que contengan las mismas palabras, como CUCHILLO GRANDE.

```
5300 REM **** S/R OBJETO VALIDO ****
5310 NN$=NN$+"  ":LN=LEN(NN$):C=1:F=0
5315 FOR K=1 TO LN
5320 IF MID$(NN$,K,1)<>"  " THEN NEXT K:RETURN
5325 W$=MID$(NN$,C,K-C):C=K+1
5330 LW=LEN(W$)
5335 FOR J=1 TO 3
5340 LI=LEN(IV$(J,1)):REM LONGITUD DEL OBJETO
5350 FOR I=1 TO LI-LW+1
5360 IF MID$(IV$(J,1),I,LW)=W$ THEN F=J:I=LI:J=3:K=LN
5370 NEXT I,J,K
5380 RETURN
```

Habiendo hallado una pareja, la rutina establece la variable F como el elemento de la matriz del inventario que corresponde al objeto de la instrucción. De no hallarse pareja, el valor de F permanece en 0, indicando que tal objeto no existe en el juego.

¿Está presente el objeto?

Una vez se ha establecido el número de matriz del objeto mediante la subrutina de prueba de validez, el escenario del objeto se puede cotejar fácilmente con la variable de escenario actual, P. El objeto a recoger está en IV$(F,1) y su situación está en IV$(F,2). La línea 3760 de la rutina RECOGER de *El bosque encantado* compara este valor con el de P. No obstante, el mensaje de error generado ("el OBJETO no está allí") podría no ser estrictamente correcto. El objeto podría estar presente en el escenario actual, pero en manos del jugador. Por consiguiente, se debe efectuar una comprobación para ver si éste transporta consigo el objeto en cuestión antes de generar el mensaje de error. De ser así, se puede producir un mensaje de error diferente (como "Tú ya tienes el OBJETO"). La siguiente subrutina verifica el inventario principal y establece un indicador, HF, a uno si el objeto lo lleva consigo el jugador. Esta condición se indica mediante un −1 en el elemento correspondiente de la matriz.

```
5450 REM **** S/R TIENE EL JUGADOR EL OBJETO ****
5460 HF=0
5470 IF IV$(OV,2)="-1" THEN HF=1
5480 RETURN
```

Comprobación de la lista

Estas dos tareas de verificar si la lista del jugador está completa y añadir a la lista se pueden combinar. Utilizando la matriz IC$() para retener los objetos transportados, se puede emplear un bucle FOR...NEXT para localizar el primer espacio libre de la matriz, de modo que se pueda entrar el nuevo objeto. Según las reglas de *El bosque encantado*, el jugador sólo puede transportar dos objetos en cualquier momento dado. Por lo tanto, el bucle FOR...NEXT utilizado sólo se ejecuta dos veces. De hallarse un espacio libre, entonces se entra el nuevo objeto; si no, sale en la pantalla un mensaje indicando que el jugador ya lleva consigo dos objetos.

La tarea final consiste en borrar el puntero de posición del objeto recién recogido del inventario. Esto se realiza poniendo IV$(F,2) a −1.

Ahora que el jugador tiene capacidad para recoger objetos, podemos incluir otra instrucción. Suele ser útil que el jugador pueda ver qué objetos está transportando. Por ejemplo, si se encuentra con una puerta cerrada con llave, puede haber olvidado que 20 movimientos antes recogió una llave. Permitir que el jugador liste los objetos que lleva consigo es una ayuda útil para la memoria. La codificación requerida es simple: se emplea un bucle FOR...NEXT para visualizar el contenido del inventario de objetos del jugador, IC$().

```
4100 REM **** LISTAR INVENTARIO TRANSPORTADO ****
4110 PRINT"OBJETOS QUE LLEVAS CONTIGO:"
4120 FOR I=1 TO 2
4130 PRINT"  ";IC$(I)
4140 NEXT I
4150 RETURN
```

Listados Digitaya

```
2140 REM **** S/R RECOGER ****
2145 IV$(4,1)="BILLETE AL TRIESTADO"
2150 GOSUB5730:REM ES VALIDO EL OBJETO
2160 IF F=0 THENPRINT"NO HAY NINGUN  ";W$:RETURN
2170 REM ** YA SE HA RECOGIDO EL OBJETO? **
2180 OV=F:GOSUB5830
2190 IFHF=1 THEN SN$="TU YA TIENES
     EL  "+IV$(F,1):GOSUB5880:RETURN
2200 :
2210 REM ** ESTA AQUI EL OBJETO **
2220 IF VAL(IV$(F,2))<>P THEN SN$=IV$(F,1)+"NO ESTA
     AQUI":GOSUB 5880:RETURN
2230 :
2240 REM ** AÑADIR OBJETO A LA LISTA **
2250 AF=0:FOR J=1TO4
2260 IF IC$(J)="THENIC$(J)=IV$(F,1):AF=1:J=4
2270 NEXT J
2280 :
2290 REM ** COMPROBAR SI CUOTA CUBIERTA **
2300 IF AF=0 THENPRINT "YA LLEVAS CUATRO OBJETOS":
     RETURN
2310 :
2320 SN$="RECOGES EL  "+IV$(F,1):GOSUB5880
2330 IV$(F,2)="-1":REM BORRAR ENTRADA POSICION
2340 RETURN

5730 REM **** S/R OBJETO VALIDO ****
5740 NN$=NN$+"  ":LN=LEN(NN$):F=0:C=1
5745 FOR K=1 TO LN
5750 IF MID$(NN$,K,1)<>"  " THEN NEXTK:RETURN
5755 W$=MID$(NN$,C,K-C):C=K+1:LW=LEN(W$)
5760 FOR J=1 TO 8
5770 LI=LEN(IV$(J,1)):REM LONGITUD OBJETO
5780 FORI=1TO LI-LW+1
5790 IFMID$(IV$(J,1),I,LW)=W$THENF=J:I=LI:J=8:K=LN
5800 NEXT I,J,K
5810 RETURN
5820 :
5830 REM **** S/R LLEVA JUGADOR EL OBJETO ****
5840 HF=0
5850 IF IV$(OV,2)="-1"THEN HF=1
5860 RETURN

2540 REM **** S/R LISTAR INVENTARIO ****
2550 PRINT"OBJETOS QUE LLEVAS CONTIGO:"
2560 FOR I=1 TO 4
2570 PRINT"  ";IC$(I)
2580 NEXT I
2590 RETURN
```

Complementos al BASIC

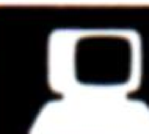
Spectrum:

En el listado de *El bosque encantado*, introduzca las siguientes modificaciones:

Sustituya SN$ por S$, IV$(,) por V$(,), IC$() por I$() y NN$ V$(,) por R$.

```
5320 IF R$(K TO K)<>"" THEN NEXT K:RETURN
5325 LET W$=R$(C TO K-1)
5360 IF V$(I TO I+LW-1)=W$ THEN LET
     F=J:LET I=LI:LET J=3:LET K=LN
```

En el listado de *Digitaya*, reemplace los nombres de las mismas variables en serie e introduzca los mismos cambios indicados arriba, pero para las líneas 5750, 5755 y 5790, respectivamente



Jugada audaz

He aquí un nuevo e interesante microordenador "de regazo": el Osborne Encore, compatible con el IBM-PC

Al haber sido el primer ordenador portátil todo en uno, el Osborne-1 marcó el inicio de una revolución en la microinformática. Equipado con un monitor incorporado, unidades de disco gemelas e interfaces para modems e impresoras, la máquina fue el primer ordenador de oficina CP/M autocontenido. Si bien la calificación de "portátil" quizá haya sido un tanto exagerada (el Osborne-1 pesaba 10,5 kg), otras firmas informáticas comprendieron muy rápidamente el potencial de la nueva máquina, y su fabricante, la empresa norteamericana Osborne Corporation, en vez de verse en un campo propio, se encontró rodeada de competidores.

En 1981 la nueva empresa recibió un duro golpe al anunciar IBM el lanzamiento del primer modelo de su gama Personal Computer. La nueva máquina enseguida barrió a todas las otras, porque los hombres de negocios se apresuraron a adquirir el modelo del conocido gigante de la industria informática. Osborne, junto con muchos de sus competidores, anunció rápidamente el inminente lanzamiento de una máquina compatible con el IBM-PC. Este nuevo modelo, el Osborne Executive, iba a estar equipado con procesadores duales, lo que le permitiría ejecutar tanto software CP/M como MS-DOS. Sin embargo, debido a la escasez mundial de microprocesadores 8086, la máquina salió al mercado sin el chip necesario para que fuera compatible con el IBM-PC. A consecuencia de ello, las ventas de los microordenadores Osborne descendieron bruscamente, y la firma norteamericana se vio obligada a presentar liquidación voluntaria en el verano de 1983. Pero Osborne logró sobrevivir y el estilo de funcionamiento de la nueva empresa es hoy similar al de Sinclair Research en Gran Bretaña, orientada hacia la investigación y el desarrollo.

Finalmente ha aparecido una nueva máquina producida por la empresa: el Osborne Encore. Es una jugada temeraria de una firma que apenas si ha conseguido permanecer en el negocio. A pesar de uno ser tan compacta como máquinas como el Epson PX-8, la importancia del Encore radica en su intento por cubrir el vacío que existe en el mercado de gestión entre el sector de máquinas "de regazo" y el de las de escritorio.

Con un peso de 6 kg, el Encore es mucho más ligero que su predecesor. El teclado se cierra contra la pantalla, conformando una caja compacta cuyo tamaño es aproximadamente el de tres listines telefónicos. La carcasa es de plástico duro de color azul y el teclado se mantiene en su sitio mediante un par de clips.

El teclado consta de teclas QWERTY tipo máquina de escribir, encima de las cuales hay una membrana plástica que comprende las teclas de función y los "iconos" (símbolos que representan a los programas incorporados). En el cuerpo principal del ordenador propiamente dicho hay una pantalla de visualización en cristal líquido que mide 23,7 × 8 cm.

El teclado ofrece un tacto sólido y profesional, con las teclas de control a ambos lados y cuatro teclas para el cursor en el extremo inferior derecho. El diseño del teclado es muy apretado y esto, sin duda, es consecuencia de incluir en un espacio tan limitado todas las características del IBM-PC. En el lado derecho del IBM hay un teclado separado que proporciona las funciones de calculadora. Para mantener la compatibilidad en el Encore, estas teclas se han incorporado en el cuerpo principal del teclado. Las funciones de calculadora están marcadas en color azul, contrastando con el etiquetado blanco de las teclas alfanuméricas estándares. A la calculadora, así como a los otros programas incorporados del Encore, se accede pulsando el icono adecuado en el panel que hay encima del teclado. Los iconos del Encore representan las rutinas de calculadora, modem, disco y calendario.

El panel sensible al tacto no está tan bien diseñado y no posee el mismo tacto profesional que ofrece globalmente el teclado. Las teclas de función (que en el IBM-PC están situadas aparte, en el sector izquierdo) poseen el mismo tipo de tacto que el teclado del ZX81. A pesar de que uno puede percibir el "pop" de la membrana de burbuja por debajo del panel, las teclas carecen de la firmeza del tacto de un teclado adecuado.

Con la incorporación de una pantalla LCD, los

Paquete compacto
El Osborne Encore es una de las primeras máquinas compatibles con el IBM-PC que viene equipada con una visualización LCD en lugar del tubo de rayos catódicos estándar. Cuando no se lo utiliza, el teclado se repliega contra la pantalla, conformando un paquete muy compacto que se puede transportar mediante la correa suministrada para colgar del hombro



Chris Stevens

diseñadores han efectuado el mayor ahorro en cuanto a consumo de energía, dado que una visualización en cristal líquido utiliza muchísima menos electricidad que el habitual tubo de rayos catódicos. Es aquí donde se presentan los problemas más grandes de compatibilidad con el IBM-PC. No es que la pantalla LCD carezca de colores, puesto que éstos se pueden sustituir fácilmente variando la tonalidad de los gráficos: es el tamaño de la pantalla lo que constituye el principal inconveniente.

La pantalla IBM normal tiene una resolución de texto de 80 por 25 caracteres; pero, debido a problemas de desarrollo que se les plantearon a los fabricantes japoneses de la pantalla para producir la visualización LCD equivalente, Osborne se ha visto obligada a introducir el Encore con una visualización de 80 por 16. Esto significa que muchos paquetes escritos para el IBM no se podrán utilizar con el Encore. Esto no representa ningún problema para los programas que desplazan sus líneas en la pantalla, pero en los paquetes en los cuales la visualización está "paginada", el usuario puede tropezar con serias dificultades, especialmente teniendo en cuenta que los mensajes aparecen por lo general en la parte inferior de la pantalla.

El lado derecho del ordenador se ha preparado para un par de unidades de disco flexibles de 5 ¼ pulgadas, si bien el modelo estándar viene equipado con una única unidad. En el lado opuesto está el conector de potencia para el transformador, un interruptor on/off, una perilla para regular el contraste de la pantalla y la caja de las pilas, que puede albergar un paquete especial de pilas de níquel-cadmio para poder hacer funcionar la máquina cuando no se tiene acceso a un enchufe eléctrico.

En la parte trasera del Encore están las puertas de E/S. De izquierda a derecha, éstas comprenden un enchufe hembra telefónico para conectar el modem incorporado del Encore, una puerta Centronics para conectar en interface con una impresora y una puerta en serie RS232 para conectar con dispositivos como impresoras en serie y modems.

Para cargar el sistema desde el disco MS-DOS basta pulsar el icono del disco en teclado sensible al tacto. Los programas incorporados se pueden ejecutar en cualquier momento, independientemente de qué programa se esté ejecutando en ese momento en la unidad de disco.

El ordenador leyó todos los discos de programas IBM que se cargaron en el Encore. No obstante, debido a las restricciones de la pantalla, resultó difícil detectar si los programas se estaban ejecutando correctamente, dado que muchas de las instrucciones se entraron "ciegas". Entre los programas que el Encore consiguió ejecutar con total éxito estaba el *Lotus 1-2-3*: un programa difícil de ejecutar en cualquier compatible debido a la forma en que accede a las rutinas incorporadas del IBM.

Resulta difícil decir si el Encore se convertirá en una máquina con tanto éxito como el Osborne-1. Acostumbrarse a la pantalla lleva algo de tiempo ya que, al igual que todas las visualizaciones LCD, necesita una luz muy intensa para producir caracteres que sean suficientemente legibles. Éste es un problema irrelevante en máquinas de regazo con caracteres grandes, pero el tamaño de los tipos del Encore es de aproximadamente la mitad, y es difícil saber si muchos usuarios querrán invertir el tiempo necesario en acostumbrarse a la visualización.

Altavoz
Las indicaciones de la pantalla se acompañan con un *beep*

Pantalla LCD
La visualización en cristal líquido permite que el Encore consuma muchísima menos potencia que una máquina provista de un tubo de rayos catódicos estándar, haciendo de esta máquina compatible con el IBM y que funciona a pilas una propuesta realista

Duplicando
El teclado del Encore posee menos teclas que el IBM-PC, pero ofrece las mismas funciones. Para conseguirlo, algunas de las teclas (señaladas con inscripciones en azul) también sirven como teclas de calculadora



Desventaja del disco

Las unidades de disco (se pueden instalar unidades de disco dobles o bien una sola) están situadas a uno de los lados de la máquina. Ello contribuye a hacer del Encore un paquete muy compacto, pero el usuario se ve forzado a estirarse para insertar un disco o comprobar si se está accediendo a la unidad adecuada



Entrada de potencia
Un transformador externo permite utilizar la energía de la red eléctrica

PCB principal
Esta placa de circuito impreso es una prueba del denso empaquetamiento que se necesita para hacer caber en un espacio tan pequeño los 512 K de memoria y otras características

Unidades de disco
Se pueden instalar unidades de disco dobles o bien una simple. La carcasa metálica alrededor de las unidades actúa a modo de disipador

Puerta para impresora
La puerta en paralelo Centronics lo conecta a una impresora

Puerta RS232
Conecta al Encore con un modem externo para comunicaciones

Chris Stevens

OSBORNE ENCORE

DIMENSIONES
241×325×141 mm

CPU
Microprocesador de 16 bits 8086

MEMORIA
Se suministra con 512 K de RAM

PANTALLA
Visualización LCD de 80×16 caracteres de 480×128 pixels. Osborne ha anunciado una pantalla de 80×25. Se les ha prometido a los usuarios una actualización económica para incorporar la pantalla más grande

INTERFACES
Interface en serie RS232, enchufe hembra telefónico RJ11 y puerta en paralelo Centronics

LENGUAJES
BASIC Microsoft en disco

TECLADO
62 teclas tipo máquina de escribir y un teclado que contiene 10 teclas de función y cuatro teclas de iconos

DOCUMENTACION
Las dos guías para el usuario están, como todos los manuales de Osborne, notablemente bien escritas; ofrece una completa explicación acerca de cómo utilizar la máquina y contiene muchísimos ejemplos

VENTAJAS
Es una máquina sumamente potente y compacta que puede operar a pilas entre cuatro y cinco horas

DESVENTAJAS
La pantalla todavía no es totalmente compatible con el IBM-PC y puede resultar difícil de leer

Control de tiempo
En la fotografía vemos la pantalla LCD de 80 × 16 caracteres. Las ranuras de debajo de la pantalla harán posible instalar una visualización de 80 × 25. También se puede observar el calendario/diario, que permite al operador entrar las futuras citas y establecer la hora en cualquiera de las zonas del mapamundi



Sensible al tacto
Encima del teclado principal hay un teclado sensible al tacto (similar al teclado del Spectrum y del ZX81) que contiene las 10 teclas de función programables. Debajo de éstas están los iconos que se utilizan para llamar a los programas basados en ROM

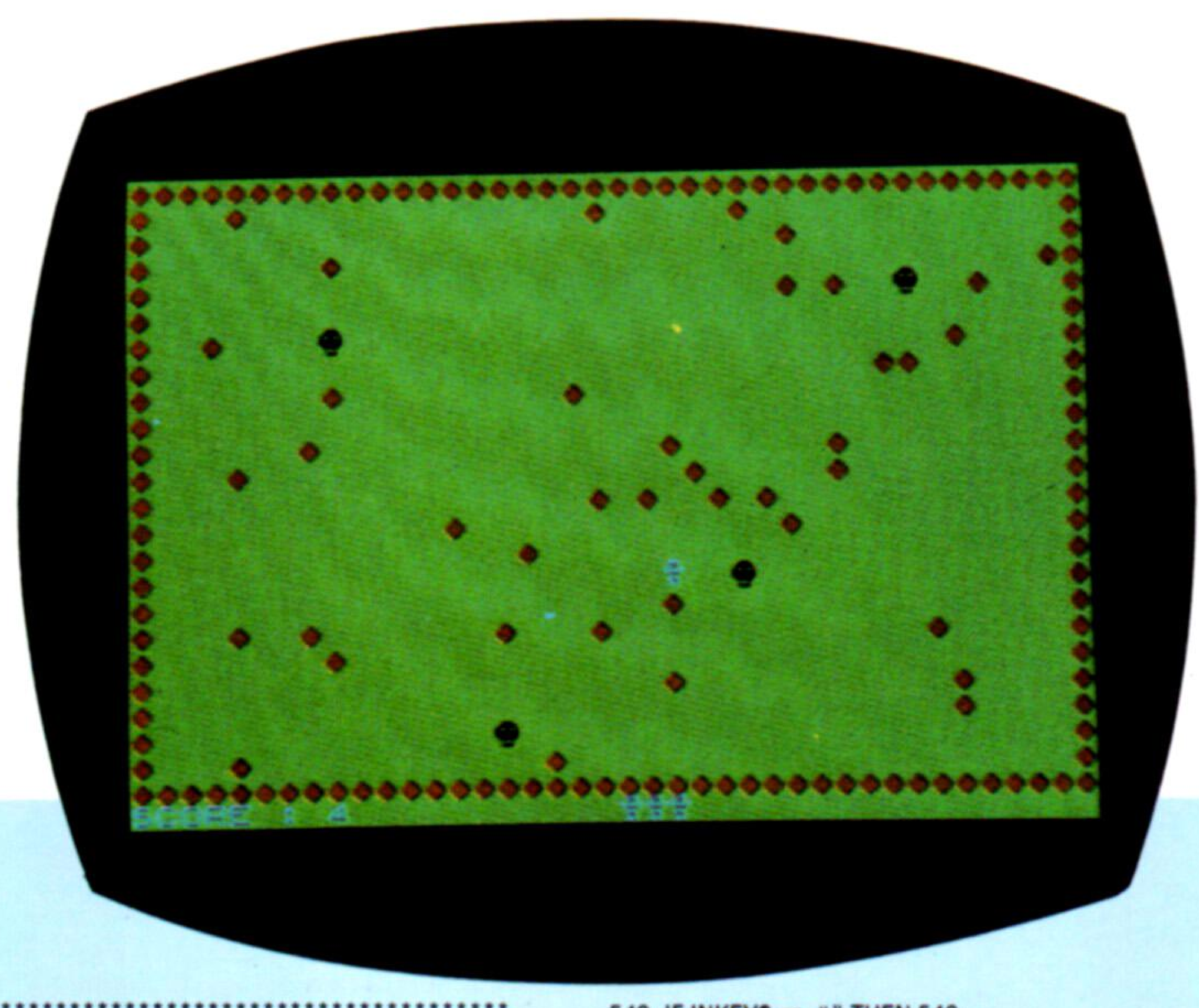
Robots

Usted se encuentra solo, abandonado en un desconocido planeta defendido por robots asesinos. He aquí un emocionante programa para el micro Thomson TO 7

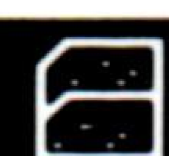
Las minas están representadas en la pantalla mediante rombos rojos. Al comenzar el juego, cinco robots se hallan sobre el terreno. Sin perder un segundo, se precipitan hacia usted, siguiendo siempre el camino más corto. Por suerte los robots son ciegos y no pueden ver las minas situadas entre usted y ellos, lo cual le permitirá, siempre que se desplace adecuadamente, eliminarlos. Para ello utilice la palanca de mando o las teclas

< A > < Z > < E >
< Q > < D >
< W > < X > < C >

según la dirección elegida por usted. Cuando haya eliminado a todos los robots, el juego continúa con un robot suplementario. Si salta sobre una mina o un robot le mata, aún no está todo perdido. En realidad tiene cinco vidas. Si desea cambiar el número de minas, modifique el valor de la variable NM en la línea 80.



```
10 REM ****************************************
20 REM *                ROBOTS                *
30 REM ****************************************
40 DEFINT A-Z
50 CLEAR ,,3
60 NH=5
70 N1=5
80 NM=40
90 NR=N1
100 DIM R(30,1)
110 GOSUB 1580
120 GOSUB 1470
130 GOSUB 910
140 ON JS GOSUB 710,810,
150 C=POINT (HX*8+4,HY*8+4)
160 IF C<>-3 AND C<>4 THEN 470
170 COLOR 4
180 LOCATE X,Y
190 PRINT N$;
200 LOCATE HX,HY
210 PRINT H$;
220 X=HX
230 Y=HY
240 T=0
250 FOR I=1 TO NR
260 IF R(I,0)=0 THEN 400
270 T=1
280 RX=R(I,0)+SGN(HX-R(I,0))
290 RY=R(I,1)+SGN(HY-R(I,1))
300 C=POINT (RX*8+4,RY*8+4)
310 IF C=1 OR C=0 THEN S=S+1:LOCATE
    R(I,0),R(I,1):PRINT N$;:R(I,0)=0:GOTO 400
320 IF C=4 THEN 470
330 COLOR 0
340 LOCATE (R(I,0),R(I,1)
350 PRINT N$;
360 LOCATE RX,RY
370 PRINT R$;
380 R(I,0)=RX
390 R(I,1)=RY
400 NEXT I
410 IF T=0 THEN 430
420 GOTO 140
430 S=S+10
440 IF INKEY$<>"" THEN 440
450 IF NR<30 THEN NR=NR+1
460 GOTO 130
470 NH=NH-1
480 COLOR 7
490 LOCATE X,Y
500 PRINT N$;
510 LOCATE HX,HY
520 PRINT H$;
530 PLAY "L96REL72REL24REL96REL72FAL24MI
    L72MIL24REL72REL24DO#L96RE"
540 IF INKEY$<>"" THEN 540
550 IF NH>0 THEN NR=N1:GOTO 130
560 CLS
570 SCREEN 1,6,6
580 ATTRB 1,1
590 LOCATE 9,10
600 PRINT "PUNTUACION ":";S;
610 LOCATE 9,20
620 PRINT "OTRA ?";
630 COLOR 4
640 ATTRB 0,0
650 IF INKEY$<>"" THEN 650
660 D$=INKEY$
670 IF D$="" THEN 660
680 IF D$<>"N" THEN RUN
690 CLS
700 END
710 D$=INKEY$
720 IF D$="A" THEN HX=HX-1:HY=HY-1
730 IF D$="Z" THEN HY=HY-1
740 IF D$="E" THEN HY=HY-1:HX=HX+1
750 IF D$="Q" THEN HX=HX-1
760 IF D$="D" THEN HX=HX+1
770 IF D$="W" THEN HX=HX-1:HY=HY+1
780 IF D$="X" THEN HY=HY+1
790 IF D$="C" THEN HY=HY+1:HX=HX+1
800 RETURN
810 J=STICK(0)
820 IF J=1 THEN HY=HY-1
830 IF J=2 THEN HY=HY-1:HX=HX+1
840 IF J=3 THEN HX=HX+1
850 IF J=4 THEN HX=HX+1:HY=HY+1
860 IF J=5 THEN HY=HY+1
870 IF J=6 THEN HY=HY+1:HX=HX-1
880 IF J=7 THEN HX=HX-1
890 IF J=8 THEN HX=HX-1:HY=HY-1
900 RETURN
910 CLS
920 COLOR 4
930 LOCATE 0,24
940 PRINT "PUNTUACION :";S;
950 IF NH=1 THEN 1000
960 FOR HX=1 TO NH-1
970 LOCATE 19+HX,24
980 PRINT H$;
990 NEXT HX
1000 COLOR 1
1010 FOR HX=0 TO 39
1020 LOCATE HX,0
1030 PRINT M$;
1040 LOCATE HX,23
1050 PRINT M$;
1060 NEXT HX
1070 FOR HY=1 TO 22
1080 LOCATE 0,HY
1090 PRINT M$;
1100 LOCATE 39,HY
1110 PRINT M$;
1120 NEXT HY
1130 FOR I=1 TO NM
1140 HX=INT(RND*38)+1
1150 HY=INT (RND*22)+1
1160 IF SCREEN(HX,HY)<>32 THEN 1140
1170 LOCATE HX,HY
1180 PRINT M$;
1190 NEXT I
1200 COLOR 0
1210 FOR I=1 TO NR
1220 R(I,0)=INT (RND*38)+1
1230 R(I,1)=INT (RND*22)+1
1240 IF SCREEN(R(I,0),R(I,1))<>32 THEN 1220
1250 LOCATE R(I,0),R(I,1)
1260 PRINT R$;
1270 NEXT I
1280 HX=INT (RND*38)+1
1290 HY=INT (RND*22)+1
1300 IF SCREEN(HX,HY)<>32 THEN 1280
1310 X=HX
1320 Y=HY
1330 FOR I=1 TO 5
1340 LOCATE HX,HY
1350 COLOR 5
1360 PRINT CHR$(127);
1370 BEEP
1380 FOR J=1 TO 50
1390 NEXT J
1400 LOCATE HX,HY
1410 COLOR 4
1420 PRINT H$;
1430 FOR J=1 TO 50
1440 NEXT J
1450 NEXT I
1460 RETURN
1470 CLS
1480 SCREEN 4,2,0
1490 ATTRB 1,1
1500 LOCATE 10,10,0
1510 PRINT "JOYSTICK ?";
1520 ATTRB 0,0
1530 D$=INKEY$
1540 C=RND
1550 IF D$="" THEN 1530
1560 IF D$="S" THEN JS=2 ELSE JS=1
1570 RETURN
1580 DEFGR$(0)=28,28,73,62,8,28,20,20
1590 DEFGR$(1)=60,126,219,255,255,126,36,60
1600 DEFGR$(2)=0,0,24,60,126,126,60,24
1610 H$=GR$(0)
1620 R$=GR$(1)
1630 M$=GR$(2)
1640 N$=CHR$(32)
1650 RETURN
```



Sesión continua

En este capítulo crearemos nuevas estructuras de control y haremos uso de las capacidades recursivas de este lenguaje

La primitiva RUN del LOGO toma una lista como entrada y hace que se ejecute como si fuera una línea de un procedimiento. Se la puede utilizar para agregarle al lenguaje nuevas estructuras de control cómo y cuándo se las requiera. De manera que podríamos definir un procedimiento MIENTRAS del siguiente modo:

```
TO MIENTRAS :CONDICION :ACCION
  IF NOT (RUN :CONDICION) THEN STOP
  RUN :ACCION
  MIENTRAS :CONDICION :ACCION
END
```

Veamos ahora un ejemplo de cómo podríamos utilizarlo. POTENCIA imprime todas las potencias de su entrada inferiores a 1000:

```
TO POTENCIA :X
  MAKE "P :X
  MIENTRAS [:P<1000][PRINT :P MAKE "P :P*:X]
END
```

Las estructuras de control, como WHILE, REPEAT y FOR, son comunes en otros lenguajes, pero en LOGO no son realmente necesarias. Una forma más natural de escribir POTENCIA sería:

```
TO POTENCIA :P
  IF NOT :P<1000 THEN STOP
  PRINT :P
  POTENCIA P*:P
END
```

No todas las versiones de LOGO disponen de REPEAT, pero esta estructura no es realmente necesaria, ya que se podría definir una palabra equivalente, REP, del siguiente modo:

```
TO REP :NUM :LISTA
  IF :NUM=0 THEN STOP
  RUN :LISTA
  REP :NUM-1 :LISTA
END
```

RUN es una primitiva sumamente útil para trabajos más avanzados. Un programa puede ensamblar una lista y pasársela luego a RUN para que se la ejecute. Enseguida veremos un ejemplo de esto.

Desarmar procedimientos

En primer lugar debemos definir un procedimiento para dibujar un triángulo de la forma habitual:

```
TO TRI
  FD 50 RT 120 FD 50
  RT 120 FD 50 RT 120
END
```

Ahora digite PRINT TEXT "TRI. El resultado será:

```
[] [FD 50] [RT 120] [FD 50] [RT 120] [FD 50] [RT 120]
```

El texto del procedimiento se da como una lista de listas, donde cada lista "interior" es una línea del procedimiento. Para ver por qué al principio hay una lista vacía, defina esta variante para la suma:

```
TO SUMAR :A :B
  PRINT :A+:B
END
```

Ahora PRINT TEXT "SUMAR dará:

```
[:A :B][PRINT :A+:B]
```

Evidentemente, la primera lista contiene las entradas para el procedimiento. De modo que TEXT nos permite introducirnos en un procedimiento y averiguar qué hay allí. DEFINE, por el contrario, hace exactamente lo inverso: nos permite definir un procedimiento como una lista de listas sin tener que acudir al editor. Pruebe ahora DEFINE "L[[:A][FD :A][RT 90][FD :A/2]] y después ejecute L utilizando, por ejemplo, L 30. El empleo de DEFINE de esta forma en modo inmediato no ofrece ninguna ventaja respecto al empleo del editor. La ventaja que nos reporta DEFINE es la posibilidad de que un procedimiento cree otro procedimiento.

Crecimiento

Ahora vamos a desarrollar un pequeño sistema para investigar el crecimiento. Las instrucciones básicas de nuestro sistema son PEDIR, que selecciona la forma de la que nos ocuparemos, y CRECER, que cambia el tamaño de la forma elegida. Por ejemplo, PEDIR "CUADRADO dibujará un cuadrado, y luego CRECER[* 10] borrará el cuadrado y volverá a dibujarlo con cada uno de sus lados incrementado en un factor de 10.

Para que los programas sean sencillos tendremos que aceptar algunas restricciones en relación al uso de estas instrucciones. En primer lugar, los procedimientos de formas dados a PEDIR como entrada no han de contener REPEAT ni llamar a subprocedimientos. En segundo lugar, el sistema se colgará si se obtienen resultados negativos. Resolver estos problemas no es muy difícil si usted desea introducir mejoras en lo que le ofreceremos aquí.

PEDIR funciona asignándole el nombre de la forma a la variable global "ACTUAL y ejecutando luego el procedimiento. Esto lo hace creando una lista de un elemento (el nombre del procedimiento) y valiéndose después de RUN para ejecutarla.

```
TO PEDIR :OBJETO
  HIDETURTLE
  MAKE "ACTUAL :OBJETO
  RUN (LIST :OBJETO)
END
```

CRECER borra primero el dibujo original (para el borrado el LOGO Commodore usa PENCOLOR-1), luego utiliza DEFINE para definir al procedimiento

Dibújame una tortuga
No se puede avanzar mucho en LOGO sin encontrarse con la recursión, algo que se define en función de sí mismo. Hemos visto ejemplos como procedimientos que se llaman a sí mismos, listas definidas en términos de listas y ahora procedimientos que escriben procedimientos. Con un poco de imaginación, en LOGO sería fácil crear un dibujo que utilizara a la tortuga para generar una tortuga que dibujara una tortuga...

actual como reescrito. El color del lápiz vuelve luego a ser normal y se dibuja la nueva forma. Observe que la entrada de CRECER se almacena en OPLIST, que nos será necesaria después.

```
TO CRECER :OPLIST
  PENCOLOR-1
  RUN (LIST :ACTUAL)
  DEFINE :ACTUAL REESCRIBIR.PROC TEXT
    :ACTUAL
    PENCOLOR 1
  RUN (LIST :ACTUAL)
END
```

REESCRIBIR.PROC separa el texto en líneas y se las va pasando de una en una a REESCRIBIR.LINEA:

```
TO REESCRIBIR.PROC :TEXTO
  IF EMPTY? :TEXTO THEN OUTPUT []
  OUTPUT FPUT REESCRIBIR.LINEA FIRST
    :TEXTO REESCRIBIR.PROC BUTFIRST :TEXTO
END
```

REESCRIBIR.LINEA busca a lo largo de una línea una FD o FORWARD. De encontrar una, le pasa el resto de la línea a CAMBIAR para que se encargue de ella.

```
TO REESCRIBIR.LINEA :LINEA
  IF EMPTY? :LINEA THEN OUTPUT []
  IF ANYOF FIRST :LINEA="FD FIRST :LINEA=
    "FORWARD THEN OUTPUT CAMBIAR BUT-
FIRST
    :LINEA
  OUTPUT FPUT FIRST :LINEA REESCRIBIR.
    LINEA BUTFIRST :LINEA
END
```

CAMBIAR construye la línea "reescrita". El primer elemento de LISTA (la entrada de CAMBIAR) habrá sido la entrada de FORWARD en el procedimiento original. Supongamos que es 50 y, si OPLIST contuviera [* 2], entonces SENTENCE FIRST :LISTA :OPLIST sería [50 * 2]. Ahora CAMBIAR utiliza RUN para evaluar esta lista (obteniendo un resultado de 100). Por último, se construye una lista compuesta por FD, la cantidad que se acaba de evaluar, y después la reescritura del resto de la línea:

```
TO CAMBIAR :LISTA
  OUTPUT(SENTENCE "FD(RUN SENTENCE
    FIRST :LISTA :OPLIST) REESCRIBIR.
    LINEA BUTFIRST :LISTA)
END
```

Imitador

En algunas ocasiones es útil poder hacer una copia de un procedimiento. De esta manera, vamos a definir un procedimiento (COPIARDEF), de modo que COPIARDEF "NUEVONOMBRE "VIEJONOMBRE defina NUEVONOMBRE como una copia de VIEJONOMBRE (VIEJONOMBRE no sufrirá ninguna alteración). Una definición obvia es:



```
TO COPIARDEF :NUEVO :VIEJO
  DEFINE :NUEVO TEXT :VIEJO
END
```

El problema de esta definición es que si VIEJO no existe, el procedimiento se limitará a seguir adelante y definirá a NUEVO como nada. Sería mejor detectar este problema e informar sobre el mismo. Por lo tanto, una definición mejor de COPIARDEF sería:

```
TO COPIARDEF:NUEVO :VIEJO
  IF NOT PROCEDIMIENTO? :VIEJO THEN
  (PRINT[NO HAY NINGUN PROCEDIMIENTO]
  :VIEJO)STOP
  DEFINE :NUEVO TEXT :VIEJO
END
```

Aquí se utiliza un procedimiento llamado PROCEDIMIENTO?, que produce VERDADERO si su entrada es un procedimiento, y, de lo contrario, FALSO. PROCEDIMIENTO? y su complemento, PRIMITIVA?, son comprobaciones muy útiles; pero lamentablemente no existen en LOGO MIT. De modo que hemos desarrollado versiones de PROCEDIMIENTO? y PRIMITIVA? que funcionarán con las versiones LOGO tanto Apple como Commodore:

```
TO PROCEDIMIENTO? :NOMBRE
  IF NUMBER? :NOMBRE THEN OUTPUT
    "FALSE IF LIST? :NOMBRE THEN OUTPUT
      "FALSE
  TEST WORD? :NOMBRE
  IFTRUE IF WORD? TEXT :NOMBRE THEN
    OUTPUT "FALSE ELSE IF NOT (TEXT
    :NOMBRE=[]) THEN OUTPUT "TRUE
    OUTPUT "FALSE
END

TO PRIMITIVA? :NOMBRE
  IF NUMBER? :NOMBRE THEN OUTPUT
    "FALSE
  IF LIST? :NOMBRE THEN OUTPUT "FALSE
  TEST WORD? :NOMBRE
  IFTRUE IF WORD? TEXT :NOMBRE THEN
    OUTPUT "TRUE ELSE OUTPUT "FALSE
END
```

Últimas palabras

Ahora ya hemos visto las características más importantes del LOGO estándar y hemos cubierto una amplia área de posibles aplicaciones. Si desea leer algo más sobre el lenguaje, a continuación le sugerimos cuatro libros:

- *Learning with LOGO*, de Daniel Watt (McGraw-Hill), es un maravilloso libro de introducción e ideal para utilizarlo con niños.
- *LOGO*, de Harold Abelson (McGraw-Hill), es el libro "estándar" sobre el lenguaje.
- *Turtle geometry*, de Harold Abelson y Andrea diSessa (MIT Press) realiza un serio análisis sobre la geometría de tortuga. La matemática involucrada es de un nivel de primer curso y de facultad; ¡en uno de los últimos capítulos se desarrolla un simulador para relatividad general en LOGO!
- *Thinking about [TLC] LOGO*, de John R. Allen, Ruth E. Davis y John F. Johnson (Holt Sanders International Editions), utiliza el TLC LOGO, que es bastante idiosincrásico, pero la obra es valiosa por sus temas de investigación y de inteligencia artificial en LOGO.

Las virtudes del LOGO

- Es interpretado, al igual que el BASIC, de modo que es fácil ejecutar y modificar programas.
- Es estructurado, con auténticos procedimientos, a diferencia de la mayor parte de las versiones de BASIC.
- Es ampliable, como el FORTH; es decir, es posible definir palabras nuevas que luego pasen a formar parte del vocabulario del ordenador.
- Posee procesamiento de listas, al igual que el LISP, por lo cual es útil para la exploración de áreas tales como la inteligencia artificial.

El LOGO en realidad no está diseñado para implementar algoritmos perfectos totalmente elaborados, pero es ideal para trabajos no especialmente "delicados". Hemos escrito la mayoría de los programas de esta serie partiendo de un procedimiento sencillo para llevar a cabo apenas una parte de la tarea. Luego alteramos el procedimiento de diversas maneras y, a consecuencia de ello, lo mejoramos y lo desarrollamos. Al final nos encontrábamos con un algoritmo perfecto bien diseñado.

Carencias del LOGO

- Las principales son el limitado espacio de trabajo y la lenta velocidad de ejecución.
- Serían útiles otras características; en particular, carece de facilidades para depuración de errores y de manipulación de matrices y archivos.

Complementos al LOGO

En todas las versiones LCSI realice las siguientes sustituciones:
NUMBERP por NUMBER?
LISTP por LIST?
WORDP por WORD?
EMPTYP por EMPTY?

El LOGO Spectrum dispone de COPYDEF (COPIARDEF) como primitiva, así como PRIMITIVEP (por PRIMITIVA?) y DEFINEDP (por PROCEDIMIENTO?).

En el Atari utilice: PC por PENCOLOR, SE por SENTENCE y HT por HIDETURTLE. DEFINE y TEXT no existen en el LOGO Atari, si bien se ofrece una forma de definirlas.

Los colores de lápiz utilizados cambiarán de una máquina a otra. Aquí se emplea PC-1 para borrar líneas, pero algunas versiones de LOGO poseen esta facilidad en la primitiva PE (de *Pen Erase*: borrar lápiz).

El interior

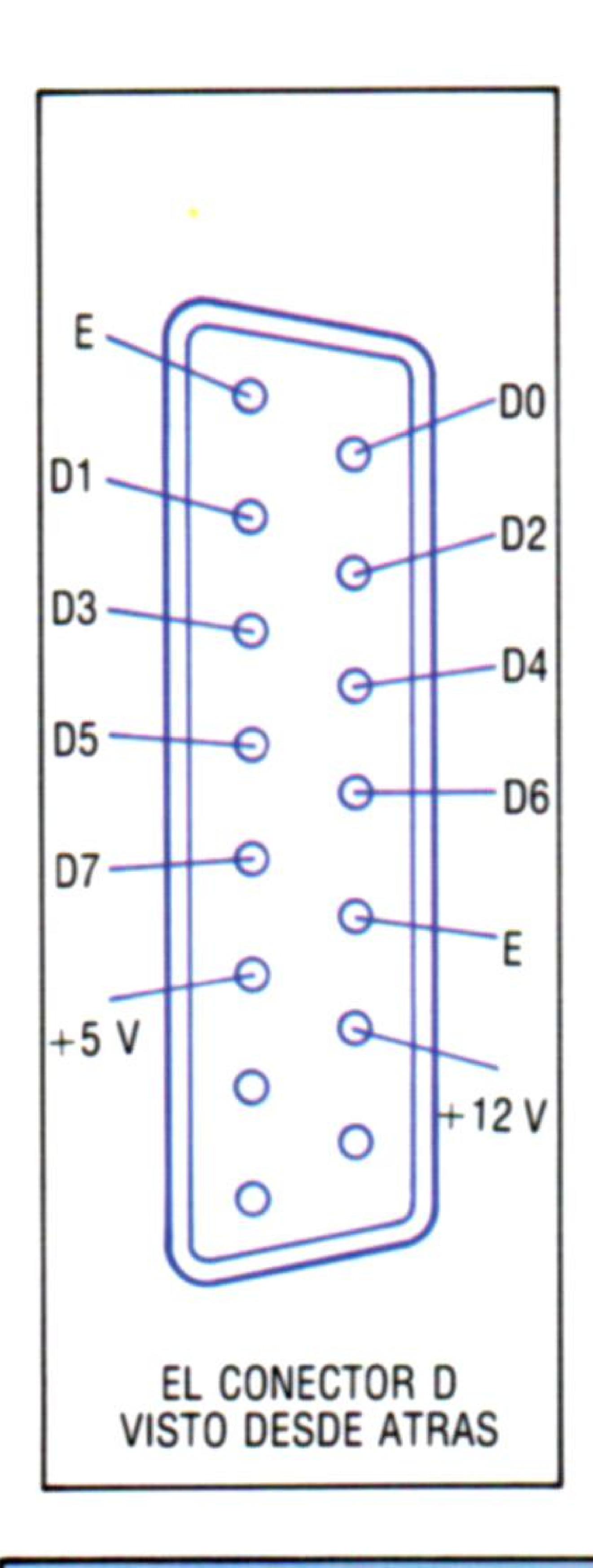
En este capítulo de nuestra serie continuaremos ensamblando el robot y escribiremos un programa que compruebe la labor que hemos realizado hasta ahora

Control del robot

Ahora que ya hemos completado la primera fase de construcción, podemos escribir un corto programa para controlar el robot desde el teclado. Los bits del 0 al 3 del registro de datos de la puerta para el usuario controlan los motores. El bit 0 es el bit de inicialización, establecido normalmente en 1; los bits 1 y 2 controlan, respectivamente, las direcciones del motor a derecha e izquierda. El bit 3 es el bit de impulso que dispara los motores para que giren otro paso. El programa utiliza las teclas T, B, F y H para controlar la dirección y un bucle repetitivo para impulsar los motores

```
1000 REM **** CONTROLADOR DEL ROBOT BBC ****
1010 RDD=&FE62:REGDAT=&FE60:?RDD=15:REM LINEAS 0-3 SALIDA
1020 PROCinicializar:REPEAT
1030 A$=INKEY$(1):IF A$<>"" THEN PROCcomprobar_teclado
1040 PROCimpulso(10)
1050 UNTIL A$="X":?REGDAT=0:END
1060 DEF PROCinicializar
1070 adelante=4:atras=2:izquierda=6:derecha=0
1080 dir=adelante:?REGDAT=dir+1:ENDPROC
1100 DEF PROCimpulso(m)
1110 FOR c=1 TO m
1120 ?REGDAT=(?REGDAT OR 8):PROCdemora(2)
1130 ?REGDAT=(?REGDAT AND 247):PROCdemora(2)
1140 NEXT c:ENDPROC
1150 DEF PROCdemora(n)
1160 FOR I=1 TO n:NEXT I
1170 ENDPROC
1180 DEF PROCcomprobar_teclado
1190 IF A$="T" THEN dir=adelante
1200 IF A$="B" THEN dir=atras
1210 IF A$="F" THEN dir=izquierda
1220 IF A$="H" THEN dir=derecha
1230 ?REGDAT=((?REGDAT AND 249)OR dir)
1240 ENDPROC
```

```
10   REM **** CONTROLADOR DEL ROBOT CMB 64 ****
20   RDD=56579:REGDAT=56577:POKERDD,15
30   GOSUB1000:REM INICIALIZAR
40   GETA$:IFA$<>"" THEN GOSUB3000:REM TECLAS
50   M=10:GOSUB1500:REM IMPULSO
60   IF A$<>"X" THEN 40
70   POKEREGDAT,0:END
1000 REM **** S/R INICIALIZAR ****
1010 AD=4:AT=2:IZ=6:DE=0
1020 DR=AD:POKEREGDAT,DR+1:RETURN
1500 REM **** S/R IMPULSO ****
1510 FOR C=1 TO M
1520 POKEREGDAT,(PEEK(REGDAT)OR8):GOSUB2000:REM DEMORA
1530 POKEREGDAT,(PEEK(REGDAT)AND247):GOSUB2000:REM DEMORA
1540 NEXT C:RETURN
2000 REM **** S/R DEMORA ****
2010 FOR I=1 TO N:NEXT I:RETURN
3000 REM **** S/R PRUEBA TECLADO ****
3010 IF A$="T" THEN DR=AD
3020 IF A$="B" THEN DR=AT
3030 IF A$="F" THEN DR=IZ
3040 IF A$="H" THEN DR=DE
3050 POKEREGDAT,((PEEK(REGDAT)AND249)ORDR)
3060 RETURN
```



EL CONECTOR D VISTO DESDE ATRAS

Conexión de enchufes

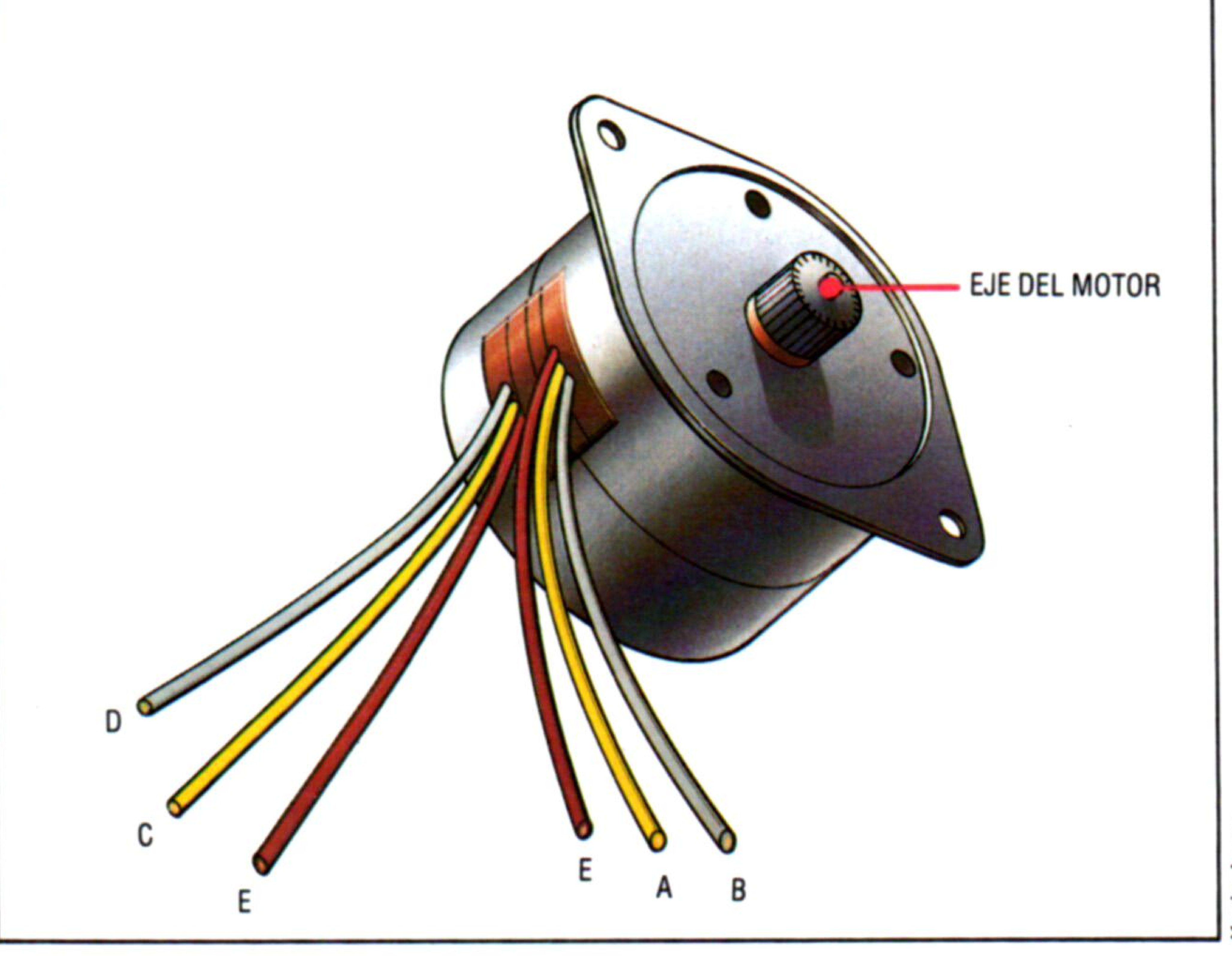
Los diagramas del enchufe de la puerta para el usuario muestran las conexiones para cada tipo de enchufe. Los usuarios del BBC Micro deben usar un cable plano de 20 vías y un conector IDC de 20 vías de ajuste a presión. Indicando las conexiones en el extremo libre del cable, se deben pelar los 11 cables necesarios y emparejar con las conexiones de la placa de interface.

Los usuarios del Commodore 64 deben usar un conector marginal de 24 vías y un trozo corto de cable plano de 12 vías. Marque las conexiones en el enchufe y empaéjelas con las conexiones de la placa. Ya que este conector se puede insertar por cualquiera de los lados, es importante señalar de alguna forma la parte superior del enchufe.

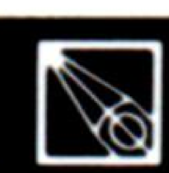
Ahora ya podemos enchufar el cable de la interface en el robot y en la puerta para el usuario. Asimismo, se debe enchufar una fuente eléctrica CD de 12 V en el conector de 2,1 mm de la placa de interface

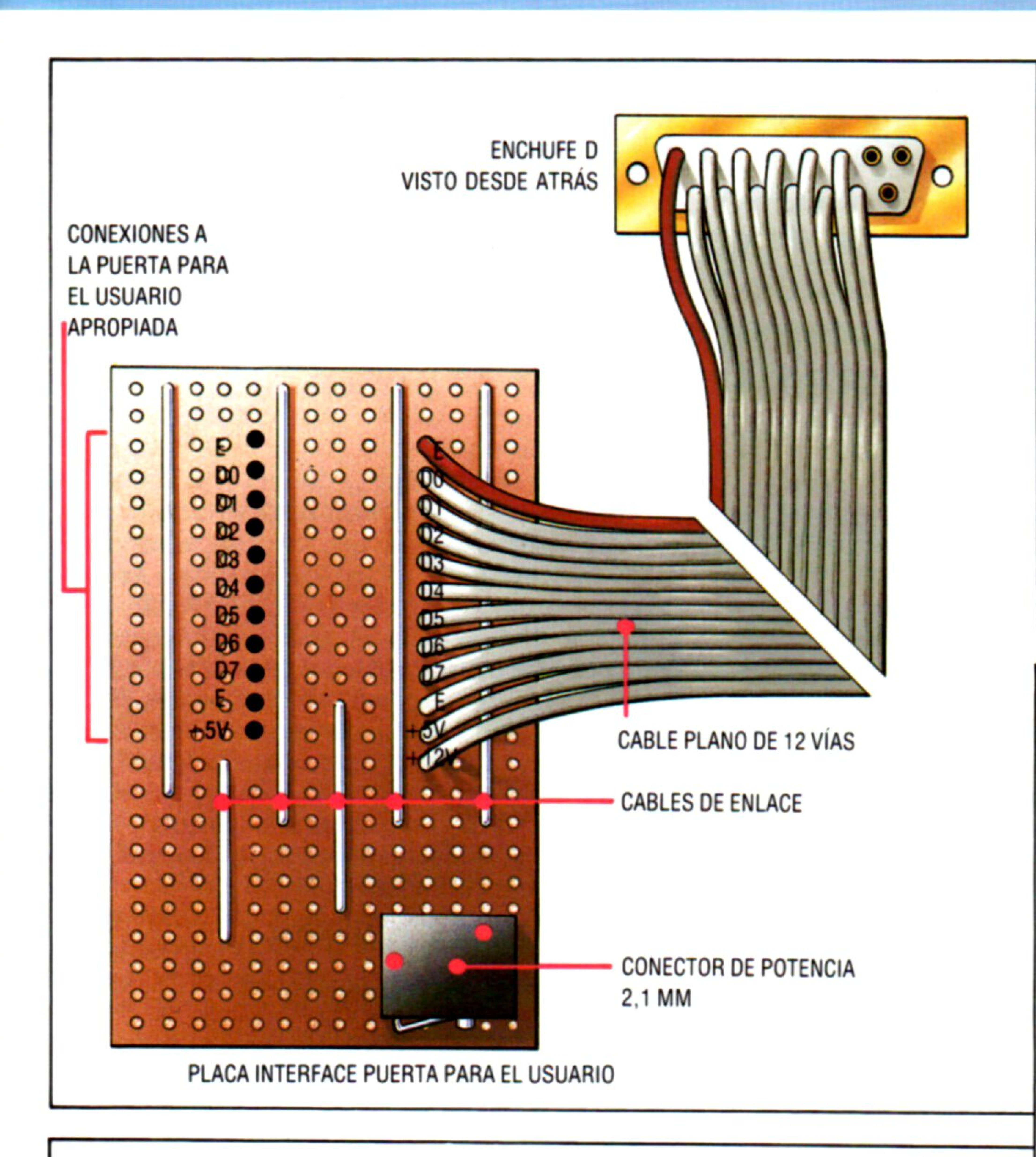
Eslabones perdidos

Para conectar los motores y el enchufe D a la placa de circuitos, tendrá que volver a remitirse a la ilustración de la placa de circuitos de p. 1317. En este diagrama vemos dónde soldar los cables correspondientes en la placa. Tome primero las conexiones del motor: cada motor tiene seis cables que salen de su cuerpo. Observe con atención que los cables emergen de la carcasa del motor en dos grupos de tres. Cada grupo de tres cables posee uno amarillo, uno gris y uno rojo. El par amarillo y gris etiquetado "A" y "B" sale de la carcasa desde el punto más cercano al eje del motor, tal como se aprecia en el diagrama. Relacionando estas letras con las conexiones con letras para cada motor en el diagrama de la placa de circuitos, suelde cada cable en su lugar en la placa



Kevin Jones





PLACA INTERFACE PUERTA PARA EL USUARIO

Lista de componentes

Cantidad	Artículo
1	Conector D 15 vías
1	Funda D 15 vías
1	Enchufe de potencia 2,1 mm
1	Conector IDC 20 vías (BBC)
1	Conector marginal 24 vías (C64)
1	Rollo parches autoadhesivos
Varios	
4 m	Cable plano 12 vías
1 m	Cable plano 20 vías (BBC)
1	Fuente alimentación CD 12 V 1 amp

Diseñar la placa

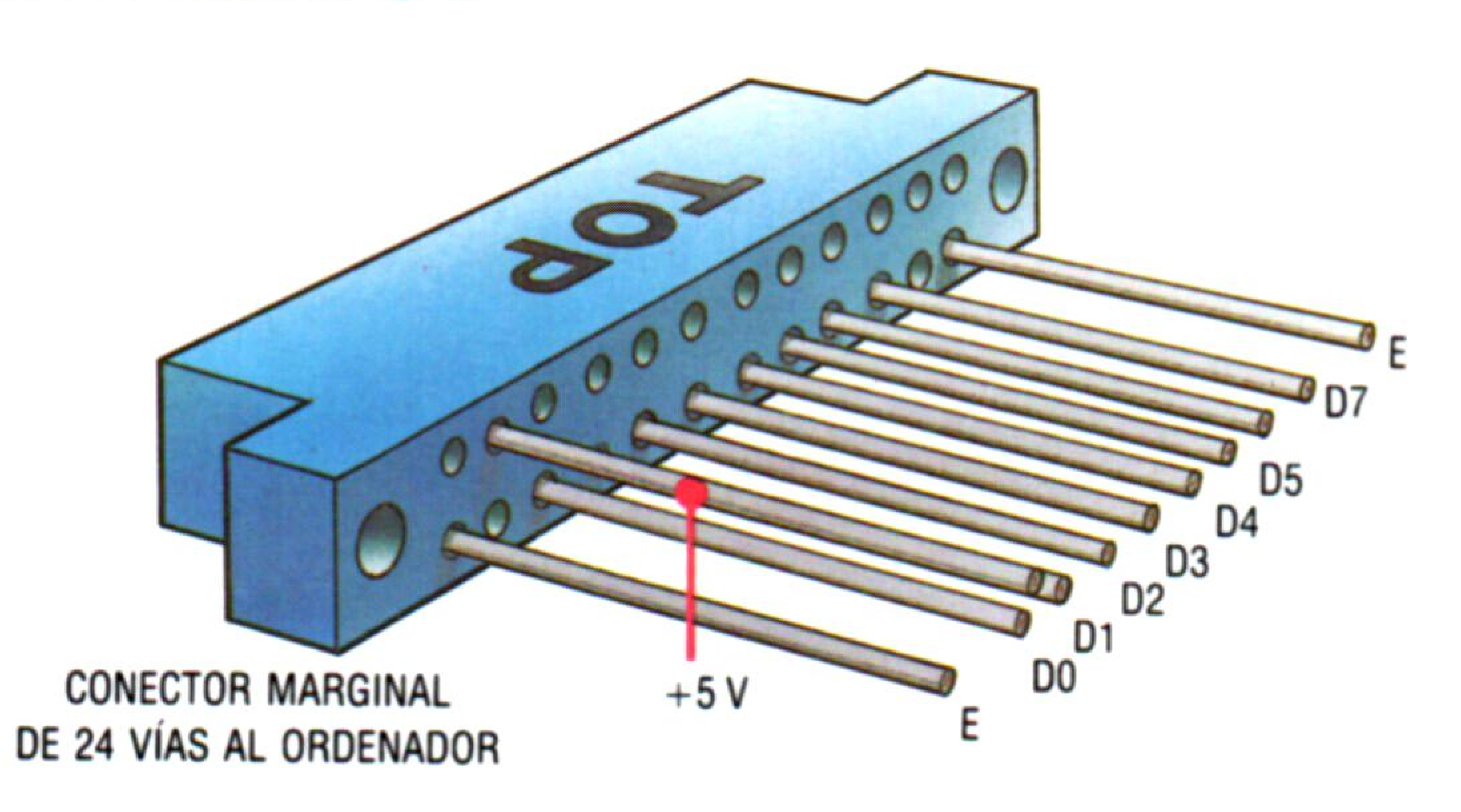
Habiendo completado las conexiones internas para el robot, hemos de diseñar una placa interface simple que nos permita controlar al robot desde la puerta para el usuario y suministrar los 12 V de CD que requieren los motores paso a paso.
Corte un trozo de veroboard de 24 franjas por 14 agujeros y conecte 3 m de cable plano de 12 vías a la placa, tal como se indica. Utilizando la tira roja de uno de los lados del cable a modo de guía, suelde los 12 cables a las patillas correspondientes de un conector D y coloque la funda del conector D para asegurar el cable.
Monte el enchufe de potencia de 2,1 mm en la placa, señalando la orientación de las patillas. El polo central de este enchufe es negativo. Luego haga los enlaces de cables tal como se indica. Observe, también, las conexiones de la puerta para el usuario; recuerde que los conectores de ésta en el BBC Micro y el Commodore 64 son diferentes

BBC Micro



CONECTOR IDC DE 20 VÍAS

Commodore 64



CONECTOR MARGINAL
DE 24 VÍAS AL ORDENADOR

Hacer conexiones

Una vez hechas las conexiones del motor, sólo nos resta hacer las conexiones adecuadas para el enchufe D, montado en la tapa del cuerpo del robot. El diagrama ilustra las conexiones de patillas correspondientes para el enchufe D, vistas desde la cara inferior de la tapa. Utilizando un trozo de cable plano de 12 vías, conecte las líneas de datos, de D0 a D3, las conexiones de +5 V, +12 V y tierra a la placa de circuitos. Deberá remitirse al diagrama de la placa de circuitos que ofrecimos anteriormente para observar las posiciones correctas del cableado de estas líneas a la placa de circuitos. Tenga especial cuidado en asegurarse de que la línea de potencia de +12 V se conecte en el punto correcto de la placa de circuitos. De no hacerlo así, se podrían dañar los circuitos internos de su ordenador. Las líneas de datos de D4 a D7 *no* se deben conectar en esta etapa, porque están reservadas para los sensores de entrada.
Todas las conexiones del interior están ahora completas. Busque dentro de la carcasa un lugar para alojar la placa de circuitos y asegúrelo con parches autoadhesivos. Cierre la tapa y fíjela con las cuatro tuercas angulares que se proporcionan

Principios operativos

Un sistema operativo hace el papel de mediador entre el programa y la máquina. Con este capítulo iniciamos el estudio de los sistemas operativos de los microordenadores más populares: BBC Micro, Sinclair Spectrum, Commodore 64 y otros

Un sistema operativo (OS: *operating system*) está escrito en el lenguaje máquina que emplea el microprocesador incorporado al ordenador. Así, por ejemplo, el OS del BBC está escrito en código máquina del 6502 y el del Spectrum en código del Z80.

Un OS se compone de una serie de rutinas a las que se confían numerosas funciones de la máquina. Según esto, puede, por ejemplo, contener una rutina que rastrea el teclado cuando se oprime una tecla, lo que permite al usuario no ocuparse de este detalle al escribir software. En un buen sistema operativo, el usuario deberá tener acceso a cualquier detalle de la máquina sin que por ello deba conocer antes su posición exacta en la memoria del ordenador o mapa de entrada/salida de la rutina que controla esa determinada parte del hardware. Esto significa que se pueden hacer más fácilmente cambios en la máquina, cuando el OS ha sido cambiado para adaptarse al nuevo hardware, de tal modo que los viejos programas puedan servir también en la nueva versión de la máquina.

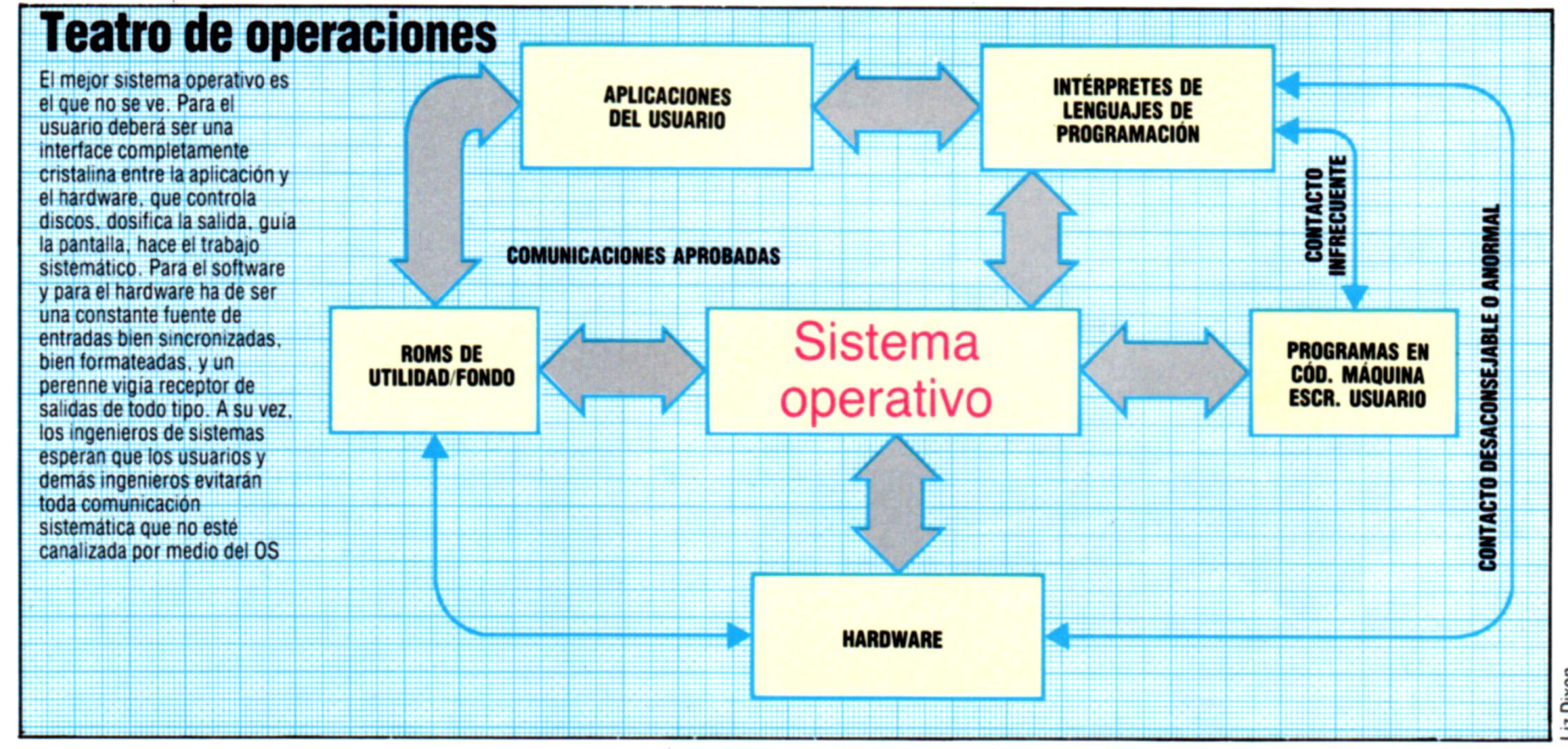
El sistema operativo del BBC, por poner un ejemplo, es un modelo de planificación. Un programa escrito en BASIC del BBC para una máquina estándar servirá perfectamente en un BBC Micro equipado con un segundo procesador, aun cuando éste represente una sustancial variación del hardware de la máquina. El OS del BBC ha atravesado varias etapas de desarrollo y experimentado diversas mejoras hasta llegar a su actual estado de refinamiento. Hace tiempo que ha desaparecido del mercado la versión 0.1, que fue la primera en aparecer. Se componía de cuatro chips EPROM, pero le faltaba flexibilidad y, lo que es más grave, no preveía las unidades de disco. La que sí lo hacía era la versión 1.0, que se componía de dos chips de 8 K colocados sobre una pequeña placa de circuito impreso dentro de la máquina. La versión actual es la 1.2, que se encuentra en la mayoría de los micros BBC utilizados hoy día. Existe una variante de esta versión, la 1.2 (US), diseñada especialmente para el mercado estadounidense.

La versión de su propia máquina la puede usted averiguar fácilmente con sólo escribir la orden *FX0 y pulsar la tecla Return. El número de la versión del OS aparecerá en pantalla. La orden *HELP también proporciona el número de la versión del OS, pero lista además los nombres de los chips ROM que posee la máquina.

Qué hace el sistema operativo del BBC

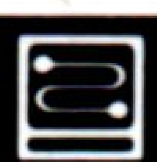
Las tareas del sistema operativo del BBC pueden agruparse en cuatro categorías principales:

1. Rutinas de entrada: Estas rutinas reciben información de la llamada *corriente de entrada en curso*.



Teatro de operaciones

El mejor sistema operativo es el que no se ve. Para el usuario deberá ser una interface completamente cristalina entre la aplicación y el hardware, que controla discos, dosifica la salida, guía la pantalla, hace el trabajo sistemático. Para el software y para el hardware ha de ser una constante fuente de entradas bien sincronizadas, bien formateadas, y un perenne vigía receptor de salidas de todo tipo. A su vez, los ingenieros de sistemas esperan que los usuarios y demás ingenieros evitarán toda comunicación sistemática que no esté canalizada por medio del OS



Ésta suele ser el teclado, pero otras corrientes de entrada son la interface RS423 y el sistema actual de archivos, al cual se accede por medio de la orden *EXEC. La mayor rutina encargada de las entradas desde las citadas corrientes se llama OSRDCH (OS *Read*D CH*aracter*: lectura de caracteres del OS).
2. *Rutinas de salida y visualización*: Encargadas de las salidas generadas por el ordenador. En el BBC hay un buen número de corrientes de salida, desde la visualización en televisor hasta la impresora, la interface RS423 y el sistema de archivos manejado por medio de la orden *SPOOL. Pero además de encargarse de la impresión y de los demás modos de dar salida a los datos, estas rutinas del OS controlan el chip de visualización 6845 dentro del ordenador y el empleo de los caracteres definidos por el usuario, por citar tan sólo dos de las funciones extra atribuidas a estas rutinas. Las llamadas del OS son las siguientes, OSWRCH (OS WR*ite* CH*aracter*: escritura de caracteres del OS), OSASCII y OSNEWL.
3. *Sistemas de archivo*: Todo sistema operativo debe proporcionar al usuario los medios para guardar el contenido de la memoria del ordenador sobre algún soporte más duradero. La sección del OS que gestiona tales transacciones se llama *sistema de archivos actuales seleccionados*, y en el BBC se tienen las opciones de la cinta magnética, el disco, el Econet, la ROM o el telesoftware. Las rutinas de archivo del OS pueden servirse de otras rutinas adicionales en ROM que instruyan al OS para el trato con el hardware asociado con el sistema particular de archivo.

Para hacer posible la interconexión con medios de almacenamiento magnético a través de ROM paginadas para sistema de archivo, se incorporan un buen número de rutinas estándar del OS destinadas a la administración de archivos. Entre ellas se encuentran rutinas para escribir o leer archivos enteros, para la obtención o el envío de bytes individuales desde un fichero abierto y rutinas de lectura o escritura de grupos de bytes desde o hacia un archivo. Las siete llamadas del OS referentes a la gestión de archivos emplean cada una un vector para indicar la rutina adecuada en el sistema de almacenamiento en cassette. Si se conecta una ROM paginada de sistema de archivo, puede aunarse con programas existentes de gestión de archivos en BASIC o en assembly cambiando simplemente estos vectores para que apunten a sus propias rutinas en ROM. Una ROM de este tipo es la DFS ROM que permite al BBC Micro el empleo de unidades de disco flexibles.
4. *Interrupciones*: En esencia, una interrupción es una señal generada sea por el hardware sea por el software que indica a la CPU que interrumpa lo que actualmente está haciendo y realice la tarea que pide inmediata atención. Una vez realizada ésta, la CPU reanuda lo que estaba haciendo como si nada hubiera ocurrido. En el BBC existen numerosas facilidades de tratamiento de interrupciones a las que el usuario puede acceder gracias al OS.

Además de estas cuatro áreas principales, existen dos llamadas del OS de capital importancia que controlan diversas funciones de la máquina. Se llaman OSBYTE y OSWORD (*word*: palabra), empleadas para controlar el chip de sonido, la tecla de interrupción o *break* y cosas similares.

¿Por qué emplear llamadas?

En la mayoría de los casos es posible obtener, a los pocos meses del lanzamiento de una máquina, la información necesaria sobre la disposición interna de la memoria del ordenador y del hardware en general (salvo implicaciones legales). Si conocemos tales detalles ¿a qué viene interesarnos por las llamadas del OS? ¿No es más sencillo acceder a los dispositivos o a la memoria directamente?

En parte ya hemos dado respuesta a esta pregunta anteriormente: las llamadas del OS nos previenen contra futuros cambios del hardware o de la configuración llevados a cabo por el fabricante. De igual modo, si se llamara a las rutinas por medio de sus actuales direcciones en la ROM nos encontraríamos con dificultades en el momento en que esta ROM fuera modificada; si se accede a las rutinas de la ROM por medio de las adecuadas llamadas del sistema operativo, se tienen siempre previstas estas eventualidades.

La instrucción que citamos ahora escribirá, en un BBC Micro común, el valor 200 en la puerta para el usuario que está en la dirección &FE60.

```
?&FE60=200
```

Si se añade un segundo procesador al ordenador, esta rutina no enviará tal valor a la puerta para el usuario, sino que lo escribirá en una posición de memoria de este segundo procesador. Por medio de una adecuada llamada del OS podemos obviar la dificultad; la llamada sabrá cómo escribir el valor en la puerta para el usuario esté o no esté conectado un segundo procesador. Tal llamada sería:

```
*FX 151,96,200
```

Se trata de una de las muchas llamadas del sistema operativo que pueden emplearse para acceder a las áreas del BBC, tales como el VIA (*versatile interface adaptor*: interface adaptador versátil para el usuario) y el bus de 1 MHz.

Por otra parte, no hay que estar siempre inventando la rueda. Si ya existe dentro de la máquina una rutina para realizar una determinada función, ¿para qué vamos a preocuparnos de ir directamente a las rutinas de la ROM? El caso es que la llamada del sistema operativo es más eficaz que el acceso directo a las rutinas implicadas en dicha llamada.

Las únicas razones realmente convincentes para acceder directamente a la memoria o a los dispositivos del hardware sería si fuera necesaria una mayor rapidez en el acceso o si no existiera la llamada del OS que realizara la tarea requerida. Si la velocidad fuera un factor crucial, el acceso directo es con frecuencia más rápido que el camino por las rutinas del OS. Sin embargo, resulta bastante peliagudo el acceso directo a la máquina, y nos guardaremos mucho de hacerlo, recordando siempre que los programas que funcionan en una máquina determinada no sirven en otras máquinas con un OS o un hardware conectado diferente. Este tipo de problemas se encuentra más en el OS del BBC, debido a los numerosos cambios que ha sufrido desde su primera aparición, que en el Spectrum, que siempre ha permanecido fiel al mismo OS. La próxima lección estará dedicada al examen de las rutinas del OS del BBC.

Más adelante trataremos del OS del Spectrum y de otros microordenadores.

Récords olímpicos

"Summer games" (Juegos de verano), original juego escrito para el Commodore 64, incluye ocho pruebas deportivas de gran emoción, desde atletismo hasta gimnasia

Summer games (Juegos de verano), de Epyx, incluye ocho pruebas deportivas, desde atletismo y natación hasta gimnasia y tiro. En cada prueba pueden participar hasta ocho competidores distintos, quienes pueden elegir un país al cual representar entre un total de 17. Los ciudadanos de países cuyas banderas no estén incluidas tienen la oportunidad de participar bajo el estandarte de Epyx.

Una vez cargado el juego, la primera escena muestra la ceremonia de apertura. Acompañado por una música convenientemente heráldica, un atleta portador de la antorcha olímpica sube corriendo hasta una plataforma y enciende la llama, mientras se libera al cielo una bandada de palomas. El sonido y los gráficos de la ceremonia de apertura son una buena muestra del conjunto del paquete. El fondo está dibujado cuidadosamente utilizando gráficos de alta resolución trazados por mapa de bits, y los uniformes movimientos del corredor y de las palomas son resultado de las excelentes facilidades de sprites del Commodore. La música, si bien no es el mejor ejemplo de lo que se puede conseguir empleando el chip SID (*sound interface device*: dispositivo interface de sonido) del Commodore 64, hace buen uso de los tres osciladores disponibles. La impresión general es que el programa, no obstante, explota al máximo las capacidades del ordenador.

Los jugadores pueden optar por participar en todas las competencias, en una sola, practicar un deporte o ver una lista de los récords mundiales de cada deporte. Hay, asimismo, una opción que permite el empleo de una o dos palancas de mando, las cuales ofrecen a los jugadores la posibilidad de competir directamente entre sí en las pruebas de natación y carrera sin necesidad de recurrir a un temporizador de paso por ordenador.

La primera de las ocho pruebas es el salto con pértiga, que quizá sea la más difícil de todas. Los jugadores compiten de uno en uno respondiendo a una serie de preguntas por parte del ordenador. Cuando está preparada la altura mínima de la barra, de 4 m, se le pregunta al jugador si desea competir a esa altura. Si la respuesta es afirmativa, el ordenador pide entonces la posición de salida de la pértiga y el atleta empieza entonces a correr a lo largo de la pantalla.

El jugador ha de tirar la palanca hacia atrás para fijar la pértiga, empujarla hacia adelante para levantar al atleta por encima de la barra y después accionar el pulsador de disparo para dejar caer la pértiga e impedir que la misma caiga contra la barra. Cada una de estas acciones exige una sincronización de fracción de segundo, puesto que un error de cálculo en cualquier punto provoca inmediatamente la caída de la barra.

Esta secuencia es una buena ilustración de la clase de parámetros que se han de considerar durante la codificación. No sólo es necesario que la pantalla esté totalmente apoyada en todo momento para asegurar un desplazamiento uniforme de los gráficos, sino que también el ordenador ha de verificar los movimientos producidos en la palanca de mando y el pulsador de disparo. Por último, el ordenador debe comprobar que la pértiga y el atleta estén en el ángulo y la posición correctos para saltar con éxito. Mientras se lleva a cabo todo ello, el usuario no debe notar que estas acciones están teniendo lugar.

Tras cada prueba aparece una tabla que muestra las medallas concedidas y se interpreta el himno nacional del país ganador. Entonces el ordenador carga la siguiente competición. El hecho de que cada prueba se cargue por separado es un indicador de la cantidad de código que se requiere para ejecutar cada prueba.

Las competiciones que siguen a continuación son salto de trampolín y gimnasia. Aquí el jugador debe manipular la palanca de mando para producir una zambullida o un salto uniforme, asegurándose de acabar con una suave entrada en el agua o de caer cuidadosamente sobre la colchoneta. El ordenador puntúa, entonces, la ejecución.

Compitiendo
La calidad del *Summer games* de Epyx se aprecia claramente en estas fotografías. Cada prueba se carga individualmente desde disco o cinta, lo que permite almacenar y procesar grandes cantidades de datos de alta resolución para los fondos

Salto de trampolín desde 15 m

Salto con pértiga

Los 100 m lisos

Ian McKinnell

Summer games: Para el Commodore 64.
Editado por: Epyx Software, 1043 Kiel Court, Sunnyvale, California, Estados Unidos.
Autores: Randy Glover, Stephen Landrum, John Leupp, Brian McGhie, Stephen Mundry, Erin Murphy, Scott Nelson
Palanca de mando: Necesaria
Formato: Versiones gemelas en cassette o disco

Editorial Delta, S.A.



10067
9 788485 822836